Institutional Repository - Research Portal
Dépôt Institutionnel - Portail de la Recherche
researchportal.unamur.be
University of Namur

THESIS / THÈSE

MASTER EN SCIENCES INFORMATIQUES

Contrôle de la qualité des données opérationnelles

Remacle, Xavier

Award date:
2009

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal ?

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Download date: 05. May. 2023

FACULTÉS UNIVERSITAIRES
NOTRE-DAME DE LA PAIX, NAMUR
INSTITUT D'INFORMATIQUE.

ANNÉE ACADÉMIQUE 2008-2009

CONTRÔLE DE LA QUALITÉ DES DONNÉES OPÉRATIONNELLES

Xavier Remacle

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION DU GRADE DE LICENCIÉ EN INFORMATIQUE.

VRS 20119877

RESUME

La qualité des données est primordiale pour la survie d'une entreprise. En effet, la réputation d'une société peut vite être compromise suite à une erreur dans ses propres données. Que penser de la confiance d'un client envers son fournisseur s'il est victime à plusieurs reprises d'erreurs de facturation ?

Le manque de qualité dans les données peut, également, avoir un impact sur les prises de décision de la société. En effet, l'intégration des données sous la forme d'information sur sa production et ses performances doit permettre à une société de prendre des décisions tactiques et stratégiques en vue d'augmenter ses bénéfices ou de se sortir d'une situation chaotique. Une entreprise a, par conséquent, plus qu'intérêt à améliorer la qualité de ses données.

L'objectif de ce mémoire est d'aborder la question de la qualité des données et de présenter une méthodologie permettant son amélioration. La méthodologie est illustrée dans une étude de cas réalisée avec l'aide de procédures développées spécifiquement pour cette démarche.

Mots-clés : qualité des données, gestion des métadonnées, standardisation, data discovery, data profiling, data matching.

ABSTRACT

The data quality is paramount for the survival of a company. Indeed, the reputation of a company can quickly be compromised following an error in its own data. What to think of the confidence of a customer towards his supplier if he is victim several times of billing error?

The lack of quality in the data can, also, have an impact on decision making for the company. Indeed, the integration of data into information on its production and performances allows a company to make tactical and strategic decisions in order to increase its benefit or to avoid a chaotic situation. Therefore a company has to improve quality of its data.

These document objectives are to tackle the data quality questions and to present a method allowing its improvement. The new method is illustrated in a case study carried out with the assistance of procedures developed specifically for the step.

Keywords: Data quality, metadata management, standardization, data discovery, data profiling, data matching.

AVANT-PROPOS

Je tiens à remercier ma famille ainsi que mes collègues de travail pour le soutien et l'encouragement qu'ils m'ont témoignés ; Madame Isabelle Boydens pour ses précieux renseignements ainsi que mon directeur de mémoire Monsieur Jean-Luc Hainaut pour l'intérêt, l'assistance, les conseils et les critiques qu'il a portés sur mes travaux.

TABLES DES MATIÈRES

TABLE DES FIGURES

Chapitre 1 : INTRODUCTION

1.1 PROBLÉMATIQUE DE LA QUALITÉ DES DONNÉES

Un système d'information doit fournir des données de qualité. En effet, les données qui y sont stockées pourront, notamment via un entrepôt de données, être croisées dans l'objectif de produire des informations permettant la prise de décisions au niveau stratégique de l'entreprise. Tout défaut de qualité dans l'information fournie pourrait amener la direction à prendre de mauvaises décisions avec des conséquences non négligeables pour l'entreprise.

Les problèmes de qualité dans les données de l'entreprise peuvent être nombreux. Ils trouvent leur origine dans les erreurs de saisie, les informations incomplètes ou encore l'absence d'implémentation des contraintes d'intégrité.

Ces problèmes de qualité sont, à notre avis, issus de deux principales raisons :

Premièrement, il est souvent difficile et coûteux de prévoir une modification du code des anciennes applications. En effet, certaines anciennes applications, datant d'avant les années 90, n'appliquaient pas correctement les standards de développement et ont été implémentées d'une façon impropre (programmation spaghetti,..). Le manque de documentation sur leurs procédures et l'impossibilité de retrouver un développeur de l'époque ou des informaticiens capables de revoir et améliorer efficacement leurs codes rendent la réingénierie de ces applications problématique.

Deuxièmement, la limitation du stockage et le coût exorbitant des disques ont orienté l'architecture informatique vers une économie maximale de l'espace disque. Ce souci d'espace a eu pour conséquences, l'historisation minimale de l'information et l'écriture d'une information trop concise, parfois même, incomplète dans le système.

1.2 PLAN DU DOCUMENT

En premier lieu, nous décrirons le concept de « Qualité » dans un système d'information. A partir de cette définition, nous allons énumérer les risques encourus par une société possédant des données pauvres en qualité. Nous chercherons, ensuite, les causes du manque de qualité dans les données.

Dans un deuxième temps, nous déterminerons une classification non exhaustive des problèmes de qualité couramment rencontrés dans les systèmes d'information. La classification sera hiérarchisée en fonction des différents niveaux d'une base de données (table, attribut et instance).

Ensuite, nous étudierons une méthodologie accompagnée de techniques et de méthodes permettant de contrôler les défauts de qualité des bases de données d'exploitation. Cette méthodologie repose sur un cycle d'amélioration de la qualité à quatre étapes. La première étape, appelée « *Data Profiling* », permettra de confronter les données actuellement dans la base de données avec leurs métadonnées respectives pour en ressortir les incohérences existantes. Ensuite, une étape de « Standardisation » permettra d'uniformiser les standards (types de données, structuration des adresses, clés primaires, clés secondaires,...) à appliquer sur les données afin d'améliorer leur qualité. De plus, la comparaison des informations sera également simplifiée grâce à une représentation des données standardisée. Une troisième étape, appelée « *Data Matching* » (recherche de doublons), permettra d'analyser les données appartenant à une même représentation de la réalité dans le but d'identifier les quelconques anomalies et incohérences. Finalement, une dernière étape, appelée « *Monitoring* », peut être définie comme un tableau de bord donnant le nombre d'anomalies nouvelles, anciennes et résolues. Chacune des trois premières étapes permettra de découvrir de nouvelles anomalies à résoudre lors d'un nouveau cycle.

Enfin, nous terminerons notre présentation par une étude de cas réalisée sur divers organismes actifs dans le domaine de la Sécurité Sociale. Cette étude débutera par une présentation des organismes de la Sécurité Sociale et de leur mission. Nous profiterons de cette présentation pour insister sur l'impact social que peut avoir une anomalie dans les données. Ensuite, différentes bases de données appartenant à ces organismes seront analysées afin de détecter si des problèmes de qualité tels que ceux présentés dans la classification y ont été découverts. Nous poursuivrons par la mise en place de procédures identifiées dans le cycle d'amélioration afin de résoudre les problèmes rencontrés.

Chapitre 2 : LA QUALITÉ DES DONNÉES

2.1 INTRODUCTION

Dans ce chapitre, nous aborderons la définition du concept de « Qualité » dans les systèmes d'information. Nous parcourrons, ensuite, les impacts que peut avoir une information de mauvaise qualité. Nous terminerons par la recherche de l'origine des problèmes de qualité.

2.2 LE CONCEPT DE QUALITÉ

La qualité d'une donnée peut être définie comme sa capacité à répondre aux objectifs qui lui sont assignés. Quatre dimensions permettent de mesurer la qualité d'une donnée [Redman, 1996] : l'exactitude, la fraîcheur, la complétude et la consistance.

L'exactitude (*Accuracy*) d'une donnée détermine son degré de précision à l'intérieur d'un ensemble de valeurs acceptables. La fraîcheur (*Currency*), quant à elle, détermine le niveau d'actualité d'une donnée. La complétude (*Completeness*) correspond au degré à partir duquel une donnée est complète, c'est-à-dire, que l'on ne peut plus rien lui ajouter pour qu'elle soit plus complète, excepté du « bruit ». Et finalement, la consistance (*Consistency*) permet de déterminer le niveau de cohérence entre les entités et les attributs.

Selon Isabelle Boydens, Professeur en Sciences de l'Information à l'ULB, la « qualité totale » n'existe pas car le concept est relatif. Sur base d'un arbitrage de type « coûts-bénéfices », les dimensions de la qualité les plus pertinentes (fraîcheur de l'information, rapidité de transmission des données, précision,...) devront être retenues dans un contexte donné. On parle de « *Fitness for use* », d'adéquation aux usages. Les besoins métiers évoluant, l'appréciation de la qualité d'une donnée ne peut donc jamais être fixée de façon définitive [Boydens, 2006].

2.3 LA QUALITÉ DES DONNÉES ET LE SYSTÈME DÉCISIONNEL

A partir des données historisées, le système décisionnel ou entrepôt de données a pour objectif d'aider les décideurs d'une entreprise à faire des choix pertinents pour la société. Ces données historisées proviennent généralement de plusieurs systèmes d'information dans lesquels les données doivent être intégrées afin de produire une information suffisamment précise et sûre pour que des actions puissent être définies en toute confiance.

Depuis quelques années, l'utilisation de l'information est devenue un instrument de survie pour l'entreprise. La vitesse à laquelle les données vont être « nettoyées », « transformées » et « intégrées » dans un entrepôt de données devient essentielle pour la compétitivité des entreprises.

2.4 IMPACT D'UNE INFORMATION PAUVRE EN QUALITÉ

Dans le monde de l'entreprise, la tendance actuelle va à l'intégration des processus métiers. Cette tendance engendre la nécessité de pouvoir accéder à une variété de systèmes d'information à l'intérieur et en dehors de l'organisation. Cependant, ces systèmes d'information peuvent contenir des données pauvres en qualité qui peuvent avoir des impacts significatifs pour l'organisation et, malheureusement pour les citoyens [Shuang-lin, 2003] (cf. Figure 2-1).

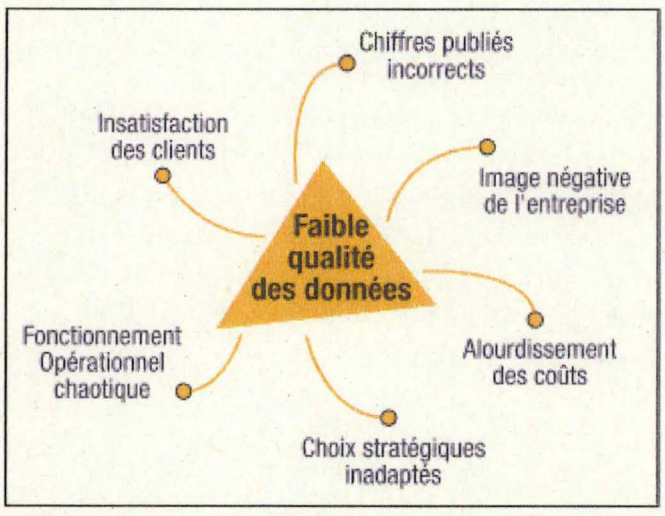

FIGURE 2-1 : PRINCIPALES CONSÉQUENCES DE LA FAIBLE QUALITÉ DES DONNÉES [BRASSEUR , 2008]

Impacts sur l'organisation :

- Diminution de la satisfaction du client : une pauvre qualité dans les données peut briser la confiance d'un client envers une organisation.

  Exemple sur la facturation : tous les clients s'attendent à avoir une facture correcte, et n'admettent pas d'erreurs sur le total d'une facture. Une erreur de ce type aura définitivement déçu les clients qui risquent de ne plus revenir.

  L'impact d'une pauvre qualité des données peut également avoir des conséquences sur les relations client dans une entreprise sans but lucratif telle qu'une bibliothèque. Par exemple, les utilisateurs d'une bibliothèque n'apprécieront pas de recevoir une réclamation de frais de retard alors qu'ils sont en ordre et que le problème provient d'une erreur du système informatique.

  Il paraît, donc, évident qu'une bonne qualité dans les données permet de garder des bonnes relations avec ses clients.

- Augmentation des coûts : une conséquence directe du problème de qualité dans les données est le gaspillage de revenu dû aux coûts opérationnels générés pour réduire les erreurs rencontrées [Redman, 1996].

  Par exemple : lors d'un vol de vacances, deux passagers se voient attribuer le même siège. Voyant les passagers s'énerver, le commandant de bord décide de donner à un des passagers un siège en première classe. Si ce genre de situation arrive souvent, il peut occasionner des coûts supplémentaires pour la compagnie aérienne.

  Un deuxième exemple : dans une bibliothèque, une erreur dans le système d'acquisition de livre peut provoquer la commande de deux livres identiques.

- Diminution dans la motivation du travail et perte de confiance dans l'organisation pour les travailleurs d'une entreprise : des problèmes de qualité dans les données diminue la satisfaction des travailleurs en leur donnant du travail supplémentaire.

  Par exemple, les employés du « service clientèle » seront largement mis sous pression s'ils reçoivent constamment des plaintes émanant de clients à propos d'erreurs de facturation.

  En revanche, des données de qualité n'améliorent pas seulement le « service clientèle », la satisfaction des clients et les relations avec ces derniers, mais améliorent aussi les performances et agrandissent la productivité interne. C'est

pourquoi, il est important d'étendre également le champ de la qualité des données des relations avec les clients externes à celles des clients internes.

En résumé, une amélioration de la qualité des données pourra accroître le moral des employés, améliorer les relations client et augmenter les marges bénéficiaires.

- Impact sur la prise de décision : comme annoncé dans le point précédent, une information de qualité est critique pour les processus de prise de décision

  Par exemple, certaines compagnies possèdent un système de gestion des risques qui surveille les risques du marché. Si les données du système sont pauvres en qualité, la direction peut, sans le savoir, être exposée à des problèmes majeurs dus à de grosses pertes de revenus.

  Dans une bibliothèque, le « département acquisition » doit surveiller constamment son budget pour ajuster correctement son plan d'acquisition. Une information incorrecte sur le statut du budget peut sérieusement avoir un impact sur le processus de prise de décision.

Impacts sociaux :

Des informations sensibles concernant des individus et des organisations, comme des enregistrements du milieu médical, financier ou judiciaire, peuvent influencer la vie des citoyens de plusieurs manières.

Par exemple, lors d'une demande pour l'ouverture d'un crédit, un banquier pourrait refuser injustement la demande suite au résultat incorrect du rapport de demande de crédit.

Un autre exemple, serait un travailleur qui se voit licencié par sa société pour avoir été fiché à tort comme étant un criminel.

Ces exemples montrent qu'une société destinée à traiter des données à caractère sensible doit s'assurer de la fiabilité et de la sécurité de ses données.

2.5 ORIGINES DES PROBLÈMES DE QUALITÉ

Les origines du manque de qualité dans les données résultent d'une culture d'entreprise et d'un contexte technique différents de ceux que nous connaissons actuellement. « *Jadis réservés aux professionnels en raison de leur complexité, de leurs exigences en puissance de calcul et de leur coût, les systèmes de gestion de bases de données sont devenus des composants de plus en plus populaires des applications informatiques de toute taille.* » [Hainaut, 2007]

Dans le passé, le coût exorbitant des machines de production et des unités de stockage ont forcé les entreprises à faire certains choix de modélisation de données dans le but de réduire les coûts au maximum. Malheureusement, avec cette focalisation sur les coûts, la facilité d'exploration et l'intégration des données n'étaient pas vues comme une priorité.

Dans les années 90, l'arrivée subite d'Internet a bouleversé le monde de l'entreprise. En effet, pour paraître à la pointe de la technologie, il fallait que l'entreprise fournisse à ses clients une solution « Web » au plus vite. Des décisions ont, par conséquent, été prises dans la précipitation. Les choix se sont souvent portés vers la création de petits systèmes d'information autonomes, c'est-à-dire possédant un système de données propre et non intégré avec les autres systèmes.

Ajoutons à cela, un comportement humain guidé par le retour d'investissement (« ROI ») provoquant une implication minime des personnes dans une documentation précise du système d'information (SI) fraîchement produit. Ce manque de documentation engendre un frein supplémentaire à la réingénierie d'une application. On se retrouve alors avec la création d'une « solution de rattrapage ». En effet, l'entreprise préférera laisser l'application tel qu'elle, et réparer les erreurs avec du post-traitement (procédures batch,...) plutôt que de se lancer dans une réingénierie des applications. En effet, cette dernière aurait nécessité l'amélioration de la documentation du système d'information imposant une exploration ardue et maladroite d'un code souvent peu structuré.

« *La sous-estimation de l'enjeu des données tient du fait que l'objectif d'une entreprise ne semble pas lié à la production des données* » [Brasseur, 2008]. Bien que l'exercice des différents métiers nécessite des informations, le but premier consiste plutôt à obtenir, selon le cas, un client satisfait, un niveau de vente supérieur, un meilleur produit, une meilleure marge,.... Les données ne se placent donc pas au premier plan de l'activité, et ne sont généralement pas perçues comme un élément essentiel de compétitivité. D'ailleurs, un grand nombre de managers considère que l'investissement dans une politique de qualité des données se traduit uniquement en perte de temps et d'argent [Redman, 1996].

Ajoutons que la mauvaise qualité des données est due principalement aux erreurs de saisie de l'information à la source. Fautes d'orthographe, codes incorrects, abréviations erronées, saisies dans un mauvais champ sont autant de sources de dégradation de la qualité qui peuvent avoir des conséquences dangereuses pour l'entreprise.

Chapitre 3 : CLASSIFICATION DES PROBLÈMES DE QUALITÉ

3.1 INTRODUCTION

Dans ce chapitre, nous allons parcourir une classification non exhaustive de différents types de problèmes qu'il est possible de rencontrer au niveau d'une base de données de production et qui serait à l'origine d'un déclin dans la qualité des données.

Nous distinguons trois catégories de problème :

Nous identifions d'abord les problèmes relatifs à un schéma. Souvent, ces problèmes sont liés à une mauvaise paramétrisation du schéma de données. Celle-ci peut concerner tant un typage laxiste des données que l'inexistence de contraintes d'intégrité référentielle.

D'autre part, nous distinguons également les problèmes liés au contenu des données elles-mêmes. Ces problèmes ne peuvent pas être détectés au niveau d'un schéma comme c'est le cas, par exemple, pour une erreur d'encodage. En effet, une valeur peut être correcte quant au type du champ, « char » par exemple, mais peut ne pas correspondre à l'information du réel qu'elle doit représenter.

Enfin, nous verrons qu'une nouvelle catégorie de problèmes peut apparaître après l'intégration de plusieurs sources de données.

Ci-dessous, la figure 3-1 permet d'illustrer la classification proposée.

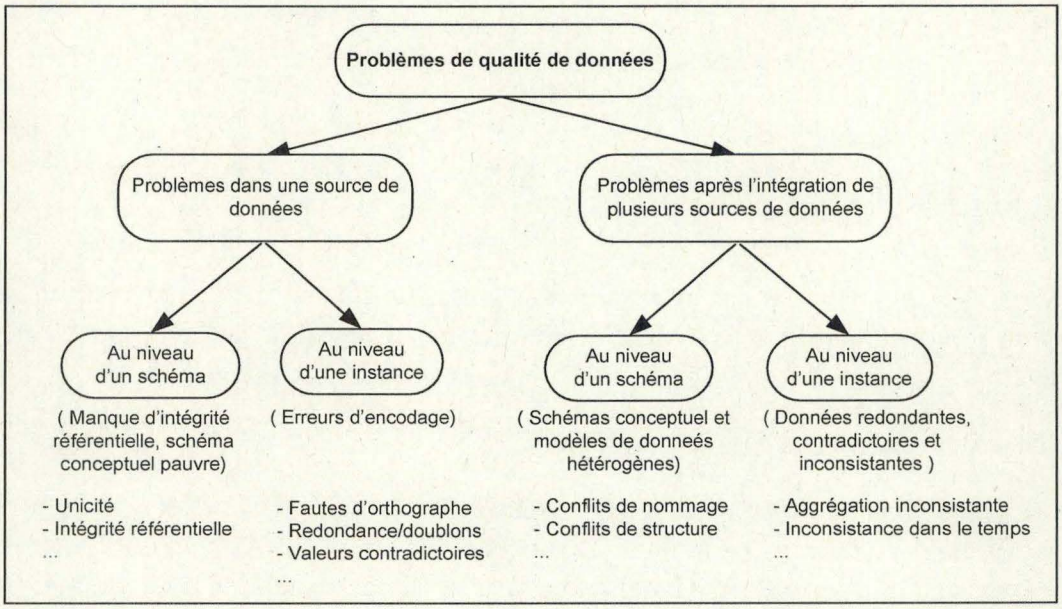

FIGURE 3-1 : CLASSIFICATION DES PROBLÈMES DE QUALITÉ DES DONNÉES

3.2 MANQUE DE QUALITÉ DANS UNE SOURCE DE DONNÉES

Nous détaillerons, dans le texte qui suit, une liste de problèmes identifiés à l'intérieur d'une source de données qui induisent un manque de qualité dans le système opérationnel.

3.2.1 LES PROBLÈMES LIÉS AUX SCHÉMAS

Les problèmes liés aux schémas sont des problèmes découlant d'une mauvaise modélisation de la base de données. Par mauvaise modélisation, nous entendons, par exemple, l'absence de contraintes d'intégrité ou la mauvaise utilisation des types de données qui permettent, dès lors, l'encodage de valeurs non autorisées dans le système de données.

Les problèmes seront présentés en quatre niveaux. Nous distinguons les problèmes au niveau d'un attribut, d'un enregistrement, d'une table et au niveau de la base donnée.

1) Au niveau d'un attribut

Les valeurs illégales

La mauvaise utilisation des types de données peut autoriser l'ajout d'une valeur illégale dans la base de données. Les valeurs illégales sont des valeurs se trouvant en dehors du domaine de validité réel de la propriété représentée par le champ.

Prenons, par exemple, une date de naissance égale à « 301370 ». Si le format du champ est défini comme « jjmmaa » alors cette date correspondra au 30/13/1970. Or, le treizième mois n'existe pas, nous avons donc une valeur erronée.

2) Au niveau d'un enregistrement

La mauvaise gestion d'un champ calculé

La mauvaise gestion d'un champ calculé peut survenir lorsque le résultat d'un champ dépend de la combinaison d'autres champs qui suivent un cycle de modification différent de celui-ci. La valeur du champ n'est donc valable que pendant une durée limitée.

Par exemple, l'attribut « Age » indique que Monsieur Dupond a 29 ans. Or, l'attribut « Date de naissance » contient comme valeur le « 01/01/1978 » et nous sommes le « 02/01/2008 ». Mr Dupond a donc aujourd'hui 30 ans, et non plus 29 ans comme indiqué dans le champ « Age ».

3) Au niveau d'une table

La violation d'unicité

Si on retrouve deux entités possédant le même identifiant, nous avons une violation d'unicité.

Par exemple, une société a enregistré deux clients ayant comme identifiant « 110 ». Après une commande d'un des deux clients, la société risque d'envoyer la facture aux deux clients.

4) Au niveau de la base de données

La violation d'intégrité référentielle

La violation d'intégrité référentielle peut survenir lorsqu'un attribut possède comme valeur une référence vers un objet de la base de données inexistant.

Citons, par exemple, le cas d'une table « Employés » qui associe le département « xxx » à un employé. Pourtant, la table des départements ne contient aucun département avec l'identifiant « xxx ».

3.2.2 LES PROBLÈMES LIÉS AUX INSTANCES

1) Au niveau d'un attribut

a. L'absence de valeur

Le fait de laisser un attribut vide (valeur « NULL ») introduit une ambiguïté sur sa signification. En effet, il est difficile de s'avoir si l'absence de valeur pour l'attribut signifie que cet attribut est non applicable ou que l'information n'est pas encore connue.

b. Les mots mal orthographiés

Les erreurs de mots mal orthographiés surviennent généralement lors d'un encodage manuel.

Ce problème peut, par exemple, se manifester lorsqu'une personne encode un mot sur une base phonétique.

Par exemple, un opérateur téléphonique encode dans un formulaire 'Rue des Ghettos' au lieu de 'Rue des Gâteaux'.

c. Les abréviations

L'utilisation d'abréviations est source d'ambiguïté et peut, par conséquent, engendrer de mauvaises interprétations.

Par exemple, l'utilisation de l'abréviation « IR » dans un attribut du domaine médical, signifie-t-il « Insuffisance Respiratoire » ou « Insuffisance Rénale » ? De plus,

l'encodage, par la suite, d' « Insuffisance Respiratoire », pourrait être interprété par le système comme une nouvelle information alors qu'il s'agit de la même.

d. Les surcharges de colonne

L'encodage de plusieurs valeurs dans une même entrée engendre un laborieux travail d'analyse syntaxique afin de pouvoir séparer les différentes valeurs de l'information. De plus, le résultat de ce travail est sans garantie d'exactitude.

Comme exemple, imaginons un champ désigné pour contenir le nom et le prénom du client mais qui contient les informations « Christian Dupond 15-01-74 Bruxelles ». La date et la valeur « Bruxelles » correspond-elle à la date et au lieu de naissance du client ?

e. Les valeurs mal attribuées

Une valeur mal attribuée peut être définie comme une valeur correcte ayant été encodée dans un mauvais attribut.

Un exemple simple est l'encodage de la ville dans l'attribut « Pays », et l'encodage du pays dans l'attribut « Ville ».

2) Au niveau d'un enregistrement

a. La violation des dépendances fonctionnelles

La dépendance fonctionnelle se définit par une dépendance forte entre des attributs, c'est-à-dire, qu'un ou plusieurs d'entre eux servent à préciser le ou les autres. Par exemple, pour une adresse, le champ « Code postal » et le champ « Ville ».

Une violation des dépendances fonctionnelles se produit lorsque deux attributs ayant une dépendance possèdent des valeurs incompatibles.

Par exemple, un premier attribut « Code postal » contient la valeur '7000' et l'attribut « Ville » contient la valeur 'Bruxelles'.

b. Les transpositions de mots

Les transpositions de mots sont possibles lorsqu'un attribut est destiné à contenir plusieurs valeurs dans un ordre prédéfini.

Prenons comme exemple, le cas d'un attribut « Nom » qui doit contenir le nom et le prénom. Il est, dès lors possible, d'encoder le nom suivi du prénom ou bien le prénom

suivi du nom. Dans ces conditions, il est difficile pour le système de distinguer le prénom du nom.

3) Au niveau d'une table

a. Les informations redondantes

Comme conséquence de l'utilisation des abréviations, de la transposition de mots ou d'une erreur d'encodage, il est possible que des informations soient redondantes. C'est le cas d'une information identique, mais syntaxiquement différente, déjà écrite dans la table.

Citons le cas d'un employé représenté deux fois dans la table des employés « Emp1 = {Nom : J. Smith, Département : 1, ...} » et « Emp2 = {Nom : John Smith, Département : 1, ...} ».

b. Les identifiants significatifs instables

Un identifiant permet de distinguer, de façon unique, une entité à l'intérieur d'un type d'entité.

« *Il n'est pas rare qu'un identifiant contienne de l'information significative. Si cette information est instable, l'identifiant le sera également* » [Hainaut, 2003]. Un tel identifiant pourrait être, par exemple, un numéro de client.

Prenons un client qui habite Bruxelles, et qui a comme identifiant la valeur 'BXL510'. Les trois premières lettres correspondant à l'abréviation de Bruxelles, que se passera-t-il si le client déménage ? Faudra-t-il changer de numéro de client ou garder l'identifiant mais avec des données inexactes ?

c. Les fusions ou éclatements d'identifiants

Il peut arriver que des objets du réel fusionnent en une seule entité ou éclatent en plusieurs entités. Dans une telle situation, la gestion des identifiants peut poser problème (par exemple, réattribution des factures à un nouveau client,...).

Si ces derniers sont mal gérés au niveau de la base de données, les fusions ou éclatements peuvent avoir un énorme impact sur la cohérence des données. En effet, il sera difficile de faire le lien entre les anciennes et les nouvelles entités.

Citons, comme exemple, deux entreprises qui fusionnent en une seule.

4) Au niveau de la base de données

a. Les mauvaises références

Les mauvaises références surviennent lorsqu'une instance référence un objet existant mais qui ne convient pas.

Citons comme exemple, un enregistrement qui associe l'employé 'J. Smith' au département '122' alors que ce dernier appartient à un autre département.

b. Les enregistrements contradictoires

Des enregistrements contradictoires surviennent lorsqu'on retrouve deux informations contradictoires pour un même objet de la réalité.

Par exemple, la table des contribuables contient l'information selon laquelle Mr Dupont est divorcé. Pourtant, dans la table, le précédent statut civil était célibataire.

c. La mauvaise gestion des dates de transaction

Une date de transaction indique à quel moment un ou plusieurs champs ont été mis à jour. Elle se trouve généralement dans la même table que les attributs auxquels elle est associée.

Cette date est importante car elle permet de contrôler le moment où le système a pris connaissance d'une certaine information. Elle est souvent utilisée dans le contexte du chargement d'un entrepôt de données, afin de charger uniquement la dernière situation de l'information à historiser.

d. Le manque d'historisation de l'information

L'écrasement d'une information par une nouvelle information peut, dans certains cas, se révéler être un problème. En effet, il est souvent nécessaire pour des questions juridiques de conserver l'historique, le cycle de vie d'une information.

e. La mauvaise gestion des métadonnées

Le manque de documentation sur le modèle de données, la structure des tables et les valeurs autorisées rend difficile la compréhension et la validation d'une information.

3.3 MANQUE DE QUALITÉ APRÈS L'INTÉGRATION DE PLUSIEURS SOURCES DE DONNÉES

Nous détaillerons, dans les lignes qui suivent, les différents types de problèmes qu'il est possible de rencontrer après l'intégration de plusieurs sources de données.

3.3.1 LES PROBLÈMES LIÉS AUX SCHÉMAS

1) Les conflits de noms

Les conflits de noms surviennent lorsqu'un même nom est utilisé pour des objets différents ou, inversement, si des noms différents sont utilisés pour des objets identiques.

Par exemple, dans une société de rencontre, la table des célibataires contient un attribut « Type » qui correspond, dans un premier système de données, à l'origine d'une personne telle que « Méditerranéen ». Cependant, dans un deuxième système, la table des célibataires contient également un attribut « Type » mais la colonne se rapporte, dans ce cas-ci, au caractère de la personne tel que « Extraverti ».

2) Les conflits de structures

Les conflits de structures se définissent par des représentations différentes d'une même information dans des sources de données différentes.

Par exemple, d'une source à l'autre, un concept est représenté par un attribut, d'un côté, et par une table, de l'autre côté. Les types de données peuvent être différents, de même que les contraintes d'intégrité, ...

3.3.2 LES PROBLÈMES LIÉS AUX INSTANCES

La majorité des problèmes liés aux instances dans une source de données (duplication d'enregistrement, enregistrements contradictoires,...) s'applique également lorsqu'on intègre plusieurs sources de données. Cependant, de nouveaux types de problèmes peuvent surgir lors de la consolidation de deux systèmes de données différents.

1) Les différences de représentation et d'interprétation

Même en ayant les mêmes noms d'attribut et la même structure de table, il peut également y avoir des représentations et des interprétations différentes pour une même information.

Imaginons un attribut « Montant » dont les montants sont, dans une source de données, exprimés en Euro et, dans une autre, exprimés en Dollar. Nous pouvons également avoir un montant exprimé en Euro dans une source, et en centimes d'Euro dans une autre.

2) Les différents niveaux d'agrégation

Les différences au niveau du degré d'agrégation de l'information apparaissent lorsque, dans une source, on donne le détail d'une information, tandis que dans une autre, on donne la même information mais sous une forme agrégée.

Par exemple, dans une source, on a le détail des ventes par produit et, dans une autre, le détail des ventes par famille de produits.

3) Les différents moments d'enregistrement de l'information

L'enregistrement des informations peut se faire à des moments différents dans le temps d'une source à l'autre.

Dans une base de données, nous avons, par exemple, les informations relatives aux ventes de la veille tandis que dans une autre source, l'information n'est enregistrée qu'à la fin de chaque semaine.

Chapitre 4 : AMÉLIORATION DE LA QUALITÉ

4.1 INTRODUCTION

Nous proposons dans ce chapitre une méthodologie permettant d'améliorer la qualité des données de l'entreprise par l'utilisation d'un processus itératif de contrôle et de transformation des données.

4.2 CYCLE D'AMÉLIORATION DE LA QUALITÉ DES DONNÉES

Comme décrit dans la section consacrée à la définition de la qualité des données (cf. chapitre 2), la qualité totale au sein d'une source de données n'existe pas car elle dépend toujours d'un choix d'arbitrage de type « coûts-bénéfices » répondant aux besoins business (métier) à un moment donné. Pour garder une qualité dans les données, il est, par conséquent, indispensable de créer un processus continu de surveillance de la qualité des données.

L'amélioration des données doit se faire progressivement car il est impensable d'améliorer toutes les données en une seule fois [Shuang-lin, 2003]. Par conséquent, une première étape consiste à déterminer les données qu'on souhaite améliorer. Afin de pouvoir effectuer ce choix, plusieurs considérations doivent être prises en compte. Une d'entre elles est l'importance des données dans la stratégie business de l'entreprise. Une seconde est l'association des données avec les problèmes business connus. Une troisième est le taux d'erreurs rencontrés dans les données. Et une dernière considération est le coût engendré par une qualité médiocre des données.

Dans les paragraphes qui suivent, nous allons parcourir les différentes étapes à prendre en compte lorsqu'une entreprise désire mettre en place un processus d'amélioration de la qualité de ses données.

Une étape initiale, appelée la « Gestion des métadonnées » (« *Metadata Management* ») est primordiale. En effet, afin de permettre à n'importe quel utilisateur du système de données d'en comprendre le contenu et ses caractéristiques, il est indispensable de diffuser

et de maintenir à jour les métadonnées de ce système. Pour ce faire, celles-ci doivent être centralisées au sein de l'entreprise. Les métadonnées ne contiennent pas uniquement la structure des données, mais doivent également décrire le « cycle de vie » d'une donnée, en d'autres mots, sa création, sa mise à jour ainsi que sa diffusion.

Comme nous l'étudierons en détail dans la suite du document, une deuxième étape, appelée « *Data Profiling* », consiste à découvrir toutes les caractéristiques d'une donnée (type, domaine de validité, ..) et à compléter les métadonnées. Le « *Data Profiling* » permet de donner une vue factuelle de la qualité d'une base de données, c'est-à-dire la documentation de son contenu, de sa structure ainsi qu'une mesure objective de sa qualité.

Ensuite, la prochaine étape nécessaire est l'étape dite de « *Standardisation* ». Elle a pour objectif d'uniformiser la structure des données, les conventions de nommage,..., de manière à pouvoir établir un modèle de données correct suivant un seul et même vocabulaire et une seule et même structure. Cette étape permet de corriger certains problèmes identifiés dans le « *Data Profiling* ». La standardisation permet de distinguer d'une manière plus aisée les liens qui peuvent exister entre les différents objets à l'intérieur d'une même source de données, ainsi qu'entre les objets issus de plusieurs sources de données.

Une quatrième étape, nommée le « *Data Matching* », permet de détecter et de traiter les doublons et les données erronées. Cette étape consiste également à s'appuyer sur des sources externes (annuaires, référentiels,...) afin de détecter des incohérences ou d'enrichir les données.

Finalement, une dernière étape, appelée l'étape de « *Monitoring* », permet de surveiller l'entièreté du processus d'amélioration en fournissant, entre autres, des rapports sur l'état de la base de données, le nombre d'erreurs détectées et corrigées,...

Ci-dessous, la représentation du cycle d'amélioration de la qualité (cf. Figure-4-1).

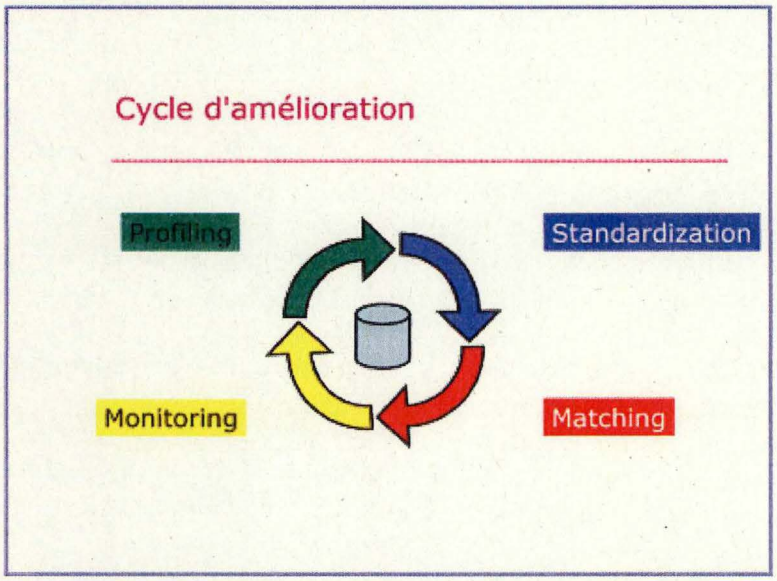

Figure 4-1 : Le cycle d'amélioration de la qualité des données [Bontemps, 2007]

4.2.1 LA GESTION DES MÉTADONNÉES

La gestion des « métadonnées » ou « méta-informations » a pour objectif de répertorier et de diffuser la documentation des applications informatiques et des modèles de données afin d'en permettre la maintenance et la réutilisation quels que soient les utilisateurs et les gestionnaires.

Les métadonnées contiennent la documentation sur les bases de données tout au long de leur cycle de vie. L'interprétation d'une même information peut varier dans le temps, mais également d'un usage à l'autre. Par conséquent, l'information doit être documentée en fonction des usages. En effet, une base de données peut être utilisée pour son objectif premier, mais peut également être mise à disposition pour répondre à d'autres demandes comme, par exemple, l'alimentation d'un entrepôt de données [Boydens, 2000].

Les métadonnées contiennent des informations complètes sur une table telles que la signification business, le type, la précision, le domaine de validité, les contraintes, le formatage,...

Elles peuvent être alimentées à partir de sources très différentes [Bontemps, 2007] (cf. Figure 4-2) :

- Les systèmes de gestion de l'information tels que les dictionnaires de données (ex : glossaires) ou les répertoires de métadonnées ;
- Les définitions de données telles que les « Copybooks COBOL », les catalogues, les schémas XML et la documentation des logiciels ;
- Les logiques et les règles métier, c'est-à-dire dans les programmes, les analyses fonctionnelles ou les instructions aux utilisateurs ;
- Le dialogue avec les administrateurs de bases de données, les architectes de données, les analystes business ou les utilisateurs.

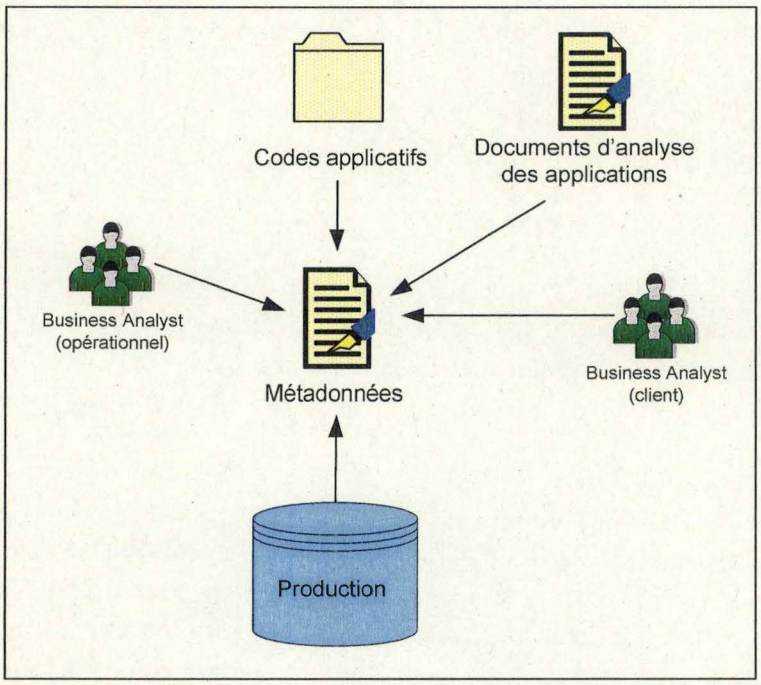

FIGURE 4-2 : SCHÉMA D'ALIMENTATION DES MÉTADONNÉES

Les recommandations pour la création des métadonnées sont les suivantes [Boydens, 2000] :

- Il est conseillé de privilégier un mécanisme d'alimentation automatique des métadonnées à partir du système. L'automatisation de l'alimentation permettra des économies en terme de mise à jour, et d'obtenir des données plus fiables ;
- Afin de répondre le plus rapidement possible au lecteur de l'information, il est recommandé de construire plusieurs niveaux de l'information en fonction des usages ;
- Il faudra désigner un administrateur du système de métadonnées qui sera chargé de l'analyse en continu du système d'information, de la mise à jour des métadonnées et de la collaboration entre les utilisateurs et les gestionnaires.

Une gestion complète et efficace des métadonnées comprendra une bonne documentation des applications existantes, ce qui permettra de mieux appréhender et de faciliter les opérations de réingénierie nécessaires pour l'amélioration de la qualité des données.

Ajoutons aussi que le fait de mieux connaître la structure des données permettra de faciliter la création de règles de détection d'anomalies dans la base de données (voir 4.2.2. Le *Data Profiling*).

Pour ce qui est du format de stockage de la documentation du modèle de données, plusieurs possibilités existent. Le langage XML, devenu un standard au niveau des échanges d'information, serait un bon choix. Les avantages du langage XML sont nombreux : il dispose d'une riche bibliothèque de types de données, une structure hiérarchique de base simple à appréhender pour le développeur et l'utilisateur final, et un standard largement reconnu par l'industrie. Il permet, par conséquent, de représenter toutes sortes de modèles de données qu'il s'agisse d'une base de données relationnelle ou d'un fichier indexé Cobol [Menet, 2006].

4.2.2 LE DATA PROFILING

Le « *Data Profiling* » consiste en une phase d'exploration du modèle de données à l'aide de techniques analytiques afin d'y découvrir la structure, le contenu et la qualité réelle des données [Olson, 2002], et d'y tester leur adéquation avec les métadonnées correspondantes.

Après analyse du résultat de l'exploration des données, des actions correctives pourront avoir lieu en vue d'adapter les métadonnées ou de détecter, puis de traiter, certaines valeurs non permises.

Le « *Data Profiling* » est une première étape indispensable dans un processus d'amélioration de la qualité, car elle permet aux gestionnaires du système d'information de se rendre compte du niveau de qualité de la base de données.

Pour arriver à ces objectifs, le « *Data Profiling* » utilise en entrée les données elles-mêmes ainsi que leurs métadonnées respectives (cf. Figure 4-3). « *Le Data Profiling suit une approche précise, méthodologique afin de parvenir, sur base des métadonnées et des données réellement présentes, dont on ne connaît en général pas la qualité, à des métadonnées corrigées et complètes d'une part, et à des informations les plus exhaustives possibles sur les données, d'autre part* » [Bontemps, 2007].

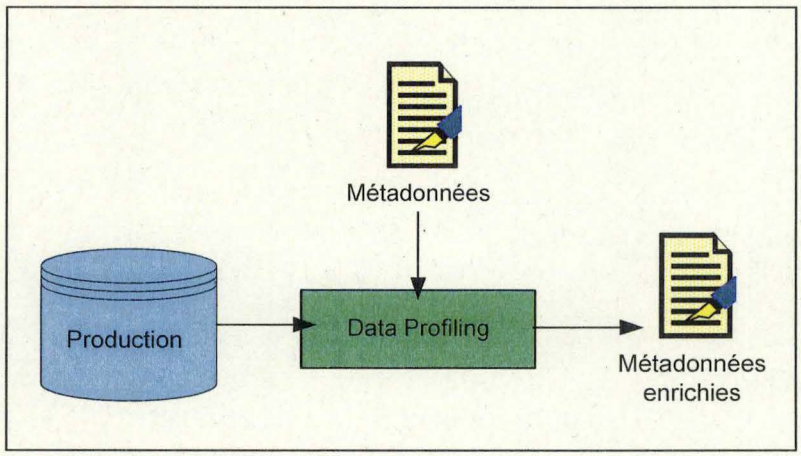

Figure 4-3 : Schéma du « *Data Profiling* »

Dans un premier temps, le « *Data Profiling* » génère des métadonnées extrêmement complètes et précises, ainsi que des informations complémentaires sur le contenu.

Une fois la définition précise des métadonnées terminée, le processus pourra, dans un deuxième temps, assurer la détection, dans les données, de la violation des contraintes décrites dans les métadonnées.

Le « *Data Profiling* » ne permettra pas de détecter toutes les données erronées, mais uniquement celles qui violent les règles établies dans les métadonnées, c'est-à-dire les valeurs invalides, les violations de structure et les violations de règles de données. Par conséquent, certaines données imprécises, ne pouvant pas être formellement vérifiées, pourront passer à travers les règles testées, et rester erronées.

Le « *Data Profiling* » se décompose en trois grandes étapes. La première est la « Préparation », la suivante l'« Analyse » et la dernière étape est la « Validation » (cf. Figure 4-4).

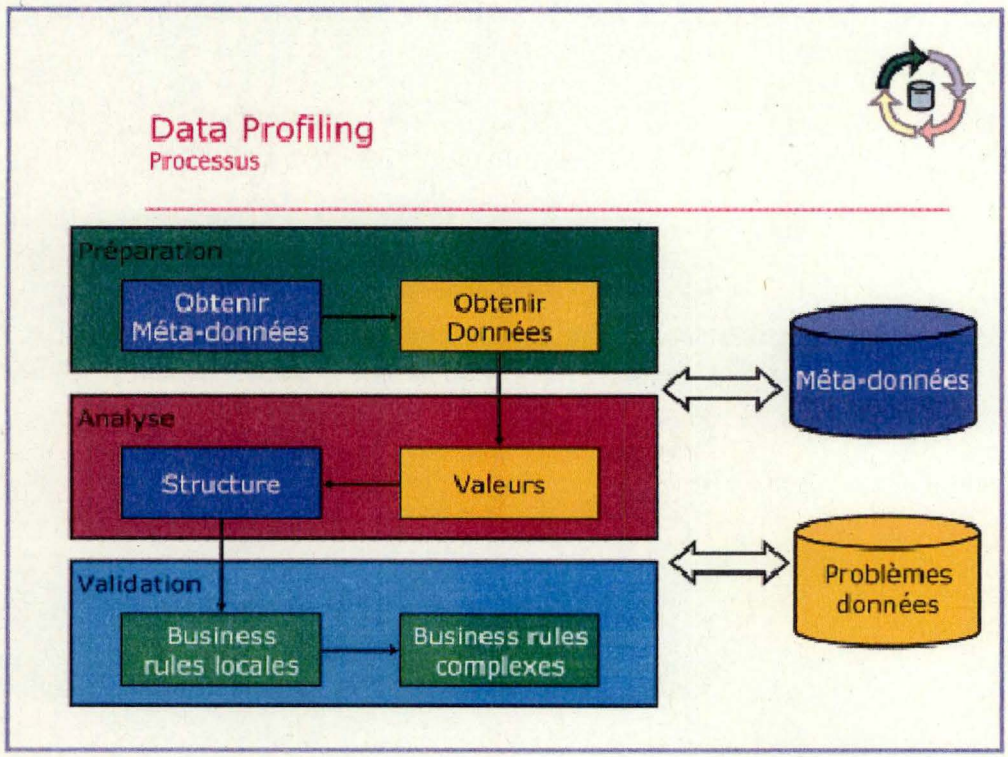

FIGURE 4-4 : LES PHASES DU DATA PROFILING [BONTEMPS, 2007]

Préparation

La première étape, la « préparation », consiste à récupérer, d'une part, les données que l'on souhaite vérifier et, d'autre part, les métadonnées nécessaires pour la vérification des données. La récupération des données est primordiale car on ne peut pas imaginer lancer un processus de vérification des données directement dans la base de données de production. Celui-ci aurait des impacts non négligeables sur les performances de la production.

Analyse

L'étape d'analyse se décompose en deux phases. Nous avons, d'une part, l'analyse de la structure de la base de données et, d'autre part, l'analyse du contenu de la base de données.

La phase d'analyse de la structure consiste à vérifier que la structure de la base de données, actuellement en production, correspond bien à sa définition dans les métadonnées. Cette phase analysera le type et la longueur des colonnes, les clés primaires, les clés étrangères,...

La phase d'analyse du contenu permet de vérifier si les valeurs attribuées aux différents attributs respectent bien les contraintes énoncées dans les métadonnées.

L'utilisation d'un outil de « *Data Profiling* » est recommandée par rapport à un développement « maison » pour les raisons suivantes [Bontemps, 2007] :

- Les algorithmes intégrés visent toujours à traiter de grandes bases de données modernes avec la plus haute performance possible ;
- Les algorithmes contiennent de nombreuses optimisations que l'on ne peut développer soi-même, à moins de disposer de connaissances mathématiques (numériques) approfondies et d'un grand nombre d'années pour le développement ;
- Un outil peut résoudre des problèmes de qualité de manière systématique et plus complète grâce à une détection automatisée, une interface utilisateur intuitive ainsi qu'un support performant.

Un outil, dédié au « *Data Profiling* » pour la phase d'analyse, permettra également de découvrir, en plus des valeurs en contradiction avec les métadonnées, la fréquence d'utilisation d'un attribut ou pour un attribut donné la fréquence et la distribution d'une valeur (cf. Figure 4-5 et Figure 4-6).

Exemple de résultat d'analyse de « *Patterns* » (=formats) faite par un outil de « *Data Profiling* » illustré par un attribut « Numero_de_telephone ». Différents « Patterns » ont été découverts dans la table analysée. Pour chacun d'entre eux, on obtiendra le nombre ainsi que le pourcentage d'enregistrements correspondant à ce « *Pattern* » :

PATTERN	COUNT	PERCENTAGE
999-999-9999	3166	96.73
(999)999-9999	42	1.28
(999) 999-9999	34	1.04
999 99 9999 999	20	0.61
999 999 9999	5	0.15
999-999-AAAA	2	0.06
9-999-999-9999	2	0.06
a	1	0.03
99 99 9999 999	1	0.03

Figure 4-5 : Exemple de résultat d'analyse de pattern [Dataflux, 2008]

Ci-dessous, un exemple de résultat d'analyse statistique faite par un outil de « *Data Profiling* ». Celui-ci reprend pour une table des informations tels que le nombre d'enregistrements uniques, la valeur minimum, la valeur maximum, le nombre de null,… :

METRIC NAME	METRIC VALUE
Data Type	double
Primary Key Candidate	no
Unique Count	1140
Uniqueness	70.11
Pattern Count	(not applicable)
Minimum Value	-223000
Maximum Value	9999999
Minimum Length	(not applicable)
Maximum Length	(not applicable)
Null Count	2
Blank Count	(not applicable)
Actual Type	double
Count	1628
Data Length	53 bit
Mean	114348.170972
Median	4888499.5
Mode	0
Non-Null Count	1626
Nullable	YES
Ordinal Position	7
Decimal Places	0
Standard Deviation	429438.361236
Standard Error	10649.778281

Figure 4-6 : Exemple de résultat d'analyse statistique [Dataflux, 2008]

Validation

L'étape de validation permet de découvrir, de formaliser, puis de documenter de nouvelles règles métier à prendre en considération dans les métadonnées, ou d'invalider des règles s'y trouvant déjà. Cette validation ne pourra se faire qu'avec les responsables métier concernés.

4.2.3 LA STANDARDISATION

Le processus de « Standardisation » est une étape importante car il permet de s'assurer que toutes les données sont encodées suivant les mêmes conventions aussi bien pour les types simples que pour les types complexes, tels que les adresses ou les noms, dans un contexte multilingue [Bontemps, 2007]. Le processus permet de modifier les données afin qu'elles utilisent les règles métier et les standards de l'entreprise [Salva, 2005]. Ces standards seront des abréviations uniformes, une orthographe correcte, des modèles de formatage,…

Les données doivent être consistantes à travers les systèmes d'information de manière à réduire les redondances d'information. Le processus de « *Data Profiling* », exécuté au préalable, permettra de faciliter la détection des données à corriger.

Ajoutons, qu'à la fin du processus de standardisation, les résultats pourront permettre de renforcer, ou de créer des nouvelles règles métier ou des nouveaux standards.

La standardisation s'opère à plusieurs niveaux : la standardisation des codes, des champs composés, des types et formats de données et des alternatives orthographiques.

- La standardisation des codes se concentre sur les codes produits, les codes financiers, les numéros d'inventaire, les numéros de modèles, les types de programmes,... Par exemple, toutes les imprimantes B/W HP doivent être encodées avec un code commençant par '*BWHP-PRN*' ce qui rendra plus facile la reconnaissance d'un produit.

- La standardisation des champs composés oblige à garder une certaine consistance dans l'ordre des différents champs de données. Un bon exemple de réorganisation de champ composé est l'adresse [Batini, 2006]. Dans la plupart des applications, l'adresse est stockée dans une seule chaîne de caractères. L'activité de standardisation pourrait être de séparer la chaîne de caractères en plusieurs sous-chaînes de caractères (parsing), comme, par exemple, le nom de la rue, le numéro, la ville et la province. Dans le contexte de l'identification d'un même objet du réel, ce type de réorganisation a pour but de rendre les comparaisons plus faciles. Un exemple de standardisation pour l'adresse *'Av. des récollets 35, Bruxelles 1000'* pourrait être *'35 AVENUE DES RECOLLETS, 1000 BRUXELLES'*.

- La standardisation des types et formats de données permet d'uniformiser leur utilisation à travers les systèmes d'information. Par exemple, '*1 Jan 2001*' ou *'01-01-2001*' doivent être adaptés vers un seul et même format.

- La standardisation des alternatives orthographiques, telles que les abréviations de mots pourrait être de les remplacer par le mot complet correspondant. Par exemple, '*Av.*' devient '*Avenue*'.

4.2.4 *LE DATA MATCHING*

L'objectif du « *Data Matching* » est de détecter les incohérences ou les doublons parmi les enregistrements, et d'en améliorer la qualité. La phase de standardisation, exécutée préalablement, a permis de préparer le terrain pour un « *Data Matching* » plus performant en permettant une comparaison plus aisée des composants concernés.

Le « *Data Matching* », appelé aussi « *Data Linkage* » ou « Appariement de données », est un processus d'identification, dans une base de données ou entre plusieurs bases de données, d'enregistrements qui définissent des mêmes entités ou objets du réel [Missier, 2003]. La décision d'associer deux enregistrements est basée sur la comparaison de caractéristiques communes, par exemple, les parties d'une adresse. La facilité de trouver des caractéristiques communes dépendra des similarités dans la représentation des données à l'intérieur des enregistrements à analyser. En effet, une information similaire est généralement représentée différemment d'une source à l'autre.

Citons le cas d'une base de données relative aux clients. A l'intérieur de celle-ci, certains clients peuvent être représentés par plusieurs enregistrements pour des raisons variées [Scannapieco, 2001] :

1) Valeurs incorrectes ou manquantes dues à des erreurs d'encodage ;
2) Utilisation de formats différents ;
3) Utilisation d'abréviations telles que '1' au lieu de 'Un' par exemple ;
4) Information incomplète suite à des données non capturées ou indisponibles ;
5) Absence de notification du changement d'adresse d'un client ;
6) Encodage de fausses informations par les clients telles qu'un faux nom ou une adresse incorrecte.

Il existe trois types de « *Matching* » ou « Appariement » :

1) One-to-One matching : le processus de « *matching* » a identifié une relation 1-1 parmi les enregistrements de deux fichiers ou bases de données.
2) One-to-Many matching : association d'un enregistrement d'une base de données avec un ou plusieurs enregistrements d'une seconde base de données.
3) Many-to-Many matching : association de chacun des enregistrements d'une base de données avec un ou plusieurs enregistrements d'une autre base de données.

Le processus de « *Data Matching* » se décompose en deux étapes : l'étape de recherche et celle de « *matching* » [Missier, 2003]. L'étape de recherche consiste à balayer les données en identifiant les enregistrements candidats pour un « *matching* ». Sur base du résultat de la première étape, le « M*atching* » va vérifier les candidats plus en profondeur, et indiquera, dans un attribut supplémentaire, la valeur '*Matched*' si l'enregistrement est identifié comme étant apparié, '*Non Matched*' s'il ne l'est pas, ou '*Possible Matched*' s'il

ne peut être formellement identifié comme '*Matched*' ou '*Non matched*'. Dans ce dernier cas, les enregistrements indiqués comme 'Possible Matched' devront être revus manuellement de manière à prendre une décision finale pour la paire d'enregistrements.

L'étape de Recherche

L'étape de Recherche se doit d'être assez intelligente afin d'exclure de la comparaison, les pairs d'enregistrements qui n'ont rien en commun. L'exécution du « *Matching* » est assez coûteuse en termes de traitements processeur. Cette première étape est cruciale car elle permet de réduire le nombre d'enregistrements à traiter lors de l'étape de « *Matching* ». Il est important de faire un compromis entre le nombre de pairs d'enregistrements à comparer et la fiabilité de la comparaison. En effet, une procédure fiable devra tester l'entièreté des combinaisons de pairs possibles, ce qui correspond à une complexité algorithmique quadratique ($O(n^2)$) si on analyse deux bases de données de même taille. La manière de résoudre ce problème est alors d'identifier seulement les pairs d'enregistrements qui ont une haute probabilité d'être appariées, laissant de côté les pairs qui semblent très différentes.

Plusieurs techniques existent dont la technique dite de « *Blocking* » [Newcombe, 1959]. Cette dernière consiste à découper l'ensemble des enregistrements en segments, appelés « *blocks* ». Chaque « *block* » contient des enregistrements ayant un ensemble de caractéristiques communes. Ces caractéristiques sont appelées les variables du « *Blocking* ». L'idée derrière le « *Blocking* » est de limiter la comparaison aux enregistrements d'un même « *block* ». Cette technique permet de diminuer drastiquement la complexité de l'étape de recherche mais peut rater des enregistrements pouvant être appariés en ne les comparant pas. De plus, il peut arriver qu'en raison de la pauvre qualité des données, les variables du « *Blocking* » soient créées à partir d'erreur dans les données avec pour conséquence, des enregistrements liés appartenant à des « *blocks* » différents.

L'étape de Matching

L'étape de « *Matching* » permet de prendre une décision définitive, concernant les candidats trouvés par l'étape de recherche, en comparant leurs caractéristiques. Habituellement les caractéristiques à comparer incluent des chaînes de caractères, pour cette raison, les algorithmes de comparaisons de chaînes de caractères jouent un rôle important dans les procédures de « *matching* » d'enregistrement. Certains types de chaînes de caractère peuvent, également, être comparés sur une base phonétique.

La méthodologie de Monica Scannapieco, professeur à l'université de Rome «La Sapienza » [Scannapieco, 2001] donne des informations plus exhaustives concernant les méthodes de « *Data Matching* ».

4.2.5 *LE MONITORING*

Le « *Monitoring* », dernière fonctionnalité du processus d'amélioration de la qualité, est incontournable car il permet de mesurer et de contrôler l'évolution du niveau de qualité des données. Il permet de vérifier si les mesures correctives ont bien été appliquées.

Le processus est automatisé et planifié et fourni entre autre des rapports sur l'état de la base de données, le nombre d'erreurs détectées et corrigées,... Ces rapports sont indispensables car ils permettront de démontrer aux décideurs de l'entreprise que l'effort mis en place pour améliorer la qualité des données porte ses fruits. Il sera, dès lors, plus facile d'avoir leur appui lors de gros changements de réingénierie qui pourrait être nécessaire.

4.3 APPLICATION DU CYCLE D'AMÉLIORATION DE LA QUALITÉ SUR LA CLASSIFICATION DES PROBLÈMES DE QUALITÉ

Dans ce troisième point, nous allons montrer que le cycle d'amélioration de la qualité des données, présenté dans ce chapitre, va permettre de détecter et de résoudre la plupart des problèmes répertoriés dans la classification des problèmes de qualité du chapitre précédent.

4.3.1 Détection et résolution des problèmes dans une source de données

1) LES PROBLÈMES LIÉS AUX SCHÉMAS

a. Au niveau d'un attribut

- *LES VALEURS ILLÉGALES*

Les valeurs illégales seront détectées lors du « *Data Profiling* », c'est-à-dire, qu'il va trouver les incohérences entre le domaine de validité du champ et la valeur présente dans l'attribut.

La phase de standardisation permettra de typer correctement les attributs de manière à interdire l'ajout d'une valeur interdite dans un champ.

b. Au niveau d'une table

- *LA VIOLATION D'UNICITÉ*

Les doublons dans une table pourront être détectés durant la phase de « *Data Matching* ».

La résolution de ce problème devra passer par la programmation du composant de l'entreprise qui a ajouté l'anomalie. Aussi, la mise en place d'une clé primaire ou secondaire correctement définie permettra d'éviter ce problème.

2) LES PROBLÈMES LIÉS AUX INSTANCES

a. Au niveau d'un attribut

Une première solution pour limiter les problèmes liés à un encodage manuel erroné est d'utiliser, dans les applications, des listes de valeurs au lieu des zones de saisie.

- *L'ABSENCE DE VALEUR*

L'absence de valeur dans un champ pourra être détectée lors du « *Data Profiling* ».

Pour résoudre ce problème, il est nécessaire de définir correctement la table en signalant à la base que l'attribut ne peut être vide (clause NOT NULL).

- *LES ERREURS DE MOTS MAL ORTHOGRAPHIÉS, LES ABRÉVIATIONS, LES SURCHARGES DE COLONNE*

Les mots mal orthographiés, les abréviations et les surcharges de colonne pourront être détectés automatiquement lors du « *Data Matching* », si des doublons apparaissent.

Ils pourront également être détectés lors du « *Data Profiling* » si leur valeur est en dehors du domaine de validité du champ.

- *LES VALEURS MAL ATTRIBUÉES*

Le « *Data Profiling* » devrait permettre de détecter les erreurs de valeurs mal attribuées si la valeur est en dehors du domaine de validité du champ.

b. Au niveau d'un enregistrement

- *LA VIOLATION DES DÉPENDANCES FONCTIONNELLES*

Le « *Data Profiling* » pourrait permettre de détecter la violation des dépendances fonctionnelles si cette dernière provoque une incohérence dans les valeurs à l'intérieur d'un enregistrement. Pour ce faire, il faut que des règles de cohérence existent au préalable dans les métadonnées. Cela signifie, plus concrètement, que, pour chaque attribut, les métadonnées devront contenir une définition complète du domaine de valeurs telle que, par exemple, attributC = attributA * attributB + 20.

c. Au niveau d'une table

- *LES INFORMATIONS REDONDANTES*

Les informations redondantes dans une table pourront être détectées durant la phase de « *Data Matching* ».

La standardisation des attributs incriminés permettra d'éviter ce problème.

- *LES IDENTIFIANTS SIGNIFICATIFS INSTABLES ET LES FUSIONS OU ÉCLATEMENTS D'IDENTIFIANTS*

Le processus d'amélioration de la qualité décrit dans ce document ne permet pas de détecter les identifiants significatifs instables ni les fusions ou éclatements d'identifiants sauf si ceux-ci donnent lieu à d'autres types d'anomalies détectables.

Pour résoudre ces types de problème, les analystes et administrateurs de données doivent s'en remettre aux règles de base de conception d'un système d'information.

- *LES ENREGISTREMENTS CONTRADICTOIRES*

Le processus de « *Data Matching* » peut être utilisé pour détecter des enregistrements contradictoires.

d. Au niveau de la base de données

- *LA MAUVAISE GESTION DES DATES DE TRANSACTION, LA MAUVAISE GESTION DES MISES À JOUR, LE MANQUE DE DOCUMENTATION SUR LES DONNÉES*

Le processus d'amélioration de la qualité n'est pas capable de détecter les manquements au niveau du fonctionnement ou de la paramétrisation d'une base de données.

3) DIAGNOSTIC DES PROBLÈMES DANS UNE SOURCE DE DONNÉES

Le tableau, ci-dessous, permet d'identifier les techniques à utiliser pour la détection, ainsi que pour la résolution de la plupart des problèmes de qualité qui peuvent intervenir dans une base de données.

Localisation	*Niveau*	*Problème*	*Détection*	*Valide*	*Solution*
Schéma	Attribut	Les valeurs illégales	Data Profiling	X	Standardisation avec un typage correct des attributs.
			Data Matching		
	Enregistrement	La mauvaise gestion d'un champ calculé	Data Profiling	X	Standardisation par l'utilisation systématique d'un champ calculé dynamiquement lors de la requête.
			Data Matching		
	Table	La violation d'unicité	Data Profiling	X	Standardisation des clés primaires et secondaires.
			Data Matching	X	
	Base de données	La violation d'intégrité référentielle	Data Profiling	X	Standardisation des clés étrangères.
			Data Matching	X	
Instance	Attribut	L'absence de valeur	Data Profiling	X	Standardisation dans l'utilisation d'une clause « *Not Null* » pour l'attribut concerné.
			Data Matching		
		Les mots mal orthographiés	Data Profiling	X	Dans certain cas, le « *Data Matching* » pourra proposer une correction.
			Data Matching	X	
		Les abréviations	Data Profiling		Standardisation des abréviations.
			Data Matching	X	
		Les surcharges de colonne	Data Profiling	X	Standardisation des formulaires d'encodage afin de prévenir les erreurs futures.
			Data Matching	X	
		Les valeurs mal attribuées	Data Profiling	X	Standardisation des formulaires d'encodage afin de prévenir les erreurs futures.
			Data Matching	X	

	Enregistrement	La violation des dépendances fonctionnelles	Data Profiling	X	Standardisation des formulaires d'encodage afin de prévenir les erreurs futures.
			Data Matching	X	
		Les transpositions de mots	Data Profiling		Standardisation des attributs composés (par exemple, les adresses).
			Data Matching	X	
	Table	Les informations redondantes	Data Profiling	X	Réingénierie des applications concernées afin de n'utiliser qu'une seule table pour l'information.
			Data Matching	X	
		Les identifiants significatifs instables	Data Profiling		Standardisation dans le choix d'une clé primaire.
			Data Matching	X	
		Les fusions ou éclatements d'identifiants	Data Profiling		Standardisation dans le choix d'une clé primaire.
			Data Matching	X	
	Base de données	Les mauvaises références	Data Profiling		Standardisation des formulaires d'encodage afin de prévenir les erreurs futures.
			Data Matching	X	
		Les enregistrements contradictoires	Data Profiling		Correction manuelle.
			Data Matching	X	
		La mauvaise gestion des dates de transaction	Data Profiling		Standardisation des tables.
			Data Matching		
		Le manque d'historisation de l'information	Data Profiling		Standardisation des mises à jour d'une table.
			Data Matching		
		La mauvaise gestion des métadonnées	Data Profiling		Standardisation des métadonnées avec une centralisation pour tous les systèmes d'information de l'entreprise.
			Data Matching		

TABLEAU 4-1 : DIAGNOSTIC DES PROBLÈMES DANS UNE SOURCE DE DONNÉES

4.4 CONCLUSION

Dans ce chapitre, nous avons pu parcourir les grandes étapes du cycle d'amélioration de la qualité.

Chaque itération permettra de détecter les nouveaux problèmes. Le « *Data Profiling* » est l'outil qui permet de détecter la plupart des problèmes en analysant aussi bien le schéma d'une base que son contenu.

Le dernier point abordé nous a montré que le processus de « Standardisation » est incontournable pour mener à bien une procédure d'amélioration de la qualité des systèmes d'information de l'entreprise. En effet, le fait de standardiser une base de données sur base de conventions uniques pour toute l'entreprise permettra à celle-ci de se documenter d'elle-même et permettra au « *Data Matching* », appliqué sur plusieurs bases de données, de fournir de meilleurs résultats lors de l'étape de « Recherche ».

Chapitre 5 : ETUDE DE CAS

5.1 INTRODUCTION

Comme nous avons pu le constater dans les chapitres précédents, les problèmes de qualité de l'information peuvent être causés par plusieurs formes de manquements ou de négligences.

Les défauts de qualité de l'information sont omniprésents dans le monde de l'entreprise. Cette mauvaise qualité touche aussi bien les entreprises du secteur privé (banques, assurances, PME, ...) que les organismes publics. Pour certains d'entre eux, le droit à l'erreur ne peut pas être admis. En effet, une erreur, même minime, peut avoir des conséquences pour l'entreprise elle-même, mais aussi, et c'est moins acceptable, des conséquences, parfois non négligeables, pour le client ou le contribuable.

Dans ce chapitre, nous illustrerons nos propos via une étude de cas portant sur des organismes de la sécurité sociale tels que l'Office National de Sécurité Sociale (ONSS), l'Office Nationale des Pensions (ONP) ou encore la Banque Carrefour de la Sécurité Sociale (BCSS).

5.2 LA SÉCURITÉ SOCIALE

5.2.1. NOTION DE SÉCURITÉ SOCIALE

En Belgique, le régime général de sécurité sociale est régi, d'une part, par la loi du 27 juin 1969 révisant l'arrêté-loi du 28 décembre 1944 concernant la sécurité sociale des travailleurs, d'autre part, par la loi du 29 juin 1981 établissant les principes généraux de la sécurité sociale des travailleurs salariés et, enfin, par les arrêtés d'exécution de ces lois [BCSS, 2008].

La sécurité sociale peut être définie comme étant l'ensemble des dispositions légales visant à garantir aux travailleurs salariés, et à leur famille, le droit à certaines prestations lorsqu'ils se retrouvent dans une situation telle qui a pour effet :

- soit de les priver de tout, ou une partie, du revenu de leur travail (par exemple, la maladie, l'invalidité, le chômage ou la retraite);
- soit de les mettre en présence de charges financières (par exemple, en matière de soins de santé ou d'éducation des enfants).

Les régimes englobés dans le système de sécurité sociale sont ceux qui concernent :

- l'assurance obligatoire en cas de maladie et d'invalidité, secteur des soins de santé;
- l'assurance obligatoire en cas de maladie et d'invalidité, secteur des indemnités;
- les allocations familiales;
- les pensions de retraite et de survie;
- l'assurance en cas de chômage;
- les vacances annuelles (des ouvriers et de certains employés);
- l'assurance en cas d'accidents du travail;
- l'assurance en cas de maladies professionnelles.

5.2.2. *TROIS RESPONSABILITÉS PRIORITAIRES*

Le rôle social de Service public, au service du Public, s'articule autour de trois responsabilités prioritaires : l'unicité de perception et de répartition des cotisations sociales (la gestion financière globale et simplifiée), l'unicité de récolte et de transmission des données administratives de base (la gestion administrative globale et simplifiée) et l'appui statistique ou documentaire en résultant [ONSS, 2007].

La première responsabilité prioritaire est la perception et la répartition correcte, régulière et à temps des cotisations sociales dues pour l'ensemble des branches du régime de sécurité sociale[1], pour le régime des vacances annuelles et pour de nombreux fonds sociaux.

[1] Les branches du régime de sécurité sociale sont les allocations familiales, les soins de santé, les indemnités d'incapacité de travail, les indemnités d'invalidité, les allocations de chômage ainsi que les pensions de retraite et de survie

La perception suppose l'instruction de plus de 220.000 déclarations personnalisées par trimestre, la gestion complexe de nombreuses formules de réductions ou d'exemptions des cotisations, ainsi que la recherche et l'identification de ceux qui ne respectent pas les règles en vigueur [ONSS, 2007].

La répartition entre les institutions publiques chargées d'octroyer les prestations sociales[2], selon les priorités et les besoins reconnus, comporte l'obligation de garantir une couverture financière complète de ces derniers, de même que le placement judicieux des moyens non utilisés [ONSS, 2007].

La deuxième responsabilité prioritaire est la récolte et la transmission correcte, régulière et à temps, pour l'ensemble des branches précitées, des données administratives requises pour instruire les droits. La récolte suppose une approche multifonctionnelle et une gestion rapide, sécurisée et interactive des données transmises en une seule fois, qui concernent principalement les salaires des travailleurs, leur carrière et leur temps de travail [ONSS, 2007].

La troisième responsabilité prioritaire est la mise à disposition, dans le respect des principes de protection de la vie privée, des données statistiques ou actuarielles dont les institutions publiques disposent et qui peuvent être utiles à l'élaboration et à l'évaluation des politiques sociales, à la recherche scientifique ou à l'information en général [ONSS, 2007].

[2] Les plus grosses institutions publiques chargées d'octroyer les prestations sociales sont l'Institut national d'assurance maladie-invalidité (I.N.A.M.I.), l'Office national d'allocations familiales pour travailleurs salariés (O.N.A.F.T.S.), l'Office national de l'emploi (O.N.Em.), l'Office national des pensions (O.N.P.), l'Office national des vacances annuelles (O.N.V.A.), le Fonds des accidents du travail (F.A.T.) et le Fonds des maladies professionnelles (F.M.P.).

5.3 LES APPLICATIONS ET LES DONNÉES OPÉRATIONNELLES

Les différentes données opérationnelles sont le plus souvent réparties dans des formes variées sur plusieurs systèmes d'information.

Différents systèmes de stockage de données sont utilisés tels que SAS, Oracle, Adabas ou Cobol.

Le système applicatif alimentant les données repose sur diverses architectures telles que Java, C, C++, Cobol, VB ou .Net.

L'information introduite concerne des millions de contribuables qui envoient leurs données sous divers formats tels que le support papier, le formulaire électronique, le fichier XML ou encore le SMS.

5.3.1. SPÉCIFICITÉS DES DONNÉES ADMINISTRATIVES

Les bases de données administratives possèdent des caractéristiques propres qui diffèrent de celles des entreprises du secteur privé.

Des modifications législatives fréquentes et complexes, dont certaines ont parfois des effets rétroactifs, exigent une gestion des versions et de l'historique [Boydens, 2006]. De plus, les informations peuvent être utilisées comme preuve lors d'un jugement, et doivent être conservées au moins jusqu'à la prescription juridique de l'information.

La gestion des données concerne des millions de contribuables répartis dans des centaines de milliers d'entreprises, ce qui représente plusieurs millions d'enregistrements ajoutés tous les mois. Ces informations souvent encodées manuellement comportent un bon nombre d'anomalies qui devront être gérées.

Tous les citoyens doivent être traités équitablement. Une simple erreur ne peut, théoriquement, pas être tolérée car cette dernière pourrait avoir une incidence sociale ou financière importante pour le citoyen.

5.4 UN ENTREPÔT DE DONNÉES

Dans le cadre de contrats d'administration conclus entre les institutions de sécurité sociale et l'Etat belge, l'institution publique a le devoir de fournir à l'Etat un ensemble d'indicateurs de performance sur l'ensemble de ses activités. Ces indicateurs ont pour but de mesurer la manière dont l'organisme public atteint ses objectifs. De plus, il n'est pas rare que des questions parlementaires soient posées directement à l'institution de sécurité sociale [ONSS, 2005].

Dans la majorité des cas, les systèmes opérationnels propres aux institutions publiques n'ont à l'origine pas été développés pour permettre des traitements statistiques. Étant donné que l'établissement de statistiques requiert généralement le traitement de très grandes quantités de données et que l'analyse statistique constitue un processus d'essais et erreurs, ces tâches sont souvent trop lourdes pour les bases de données d'exploitation. Ce problème peut, alors, être résolu par l'utilisation d'un entrepôt de données [ONSS, 2005].

Le but de l'entrepôt de données est de pouvoir répondre de manière plus correcte, plus rapide et moins onéreuse aux demandes de données. Il regroupe un ensemble de technologies complexes permettant, d'une part, de mettre à disposition un outil destiné à suivre et améliorer les performances de l'organisme de sécurité sociale et, d'autre part, de proposer un outil de gestion utilisé directement par le personnel (direction, personnel informatique, ressources humaines, chefs de service,...).

Les données nécessaires issues des systèmes opérationnels sont collectées et historisées dans un entrepôt de données. Ensuite, elles sont transformées en information et en indicateurs qui sont restitués dans des rapports de suivi. Ce sont eux qui permettent de déterminer si l'organisme de sécurité sociale a atteint ses objectifs.

L'entrepôt permet entre autres :

- de disposer de tableaux de bord efficaces et conviviaux pour l'analyse des données ainsi que des indicateurs de performance.
- de tracer l'information des travailleurs au cours du temps à travers le cycle complet des différents processus administratifs (identification, contrôle de la déclaration, situation de compte et procédures de recouvrement, inspection,...).

Etant donné, la portée politique et sociale de l'information émise par les entrepôts de données, il est primordial que les données utilisées pour créer l'information demandée soient correctes et de bonne qualité. En effet, les décisions qui seront prises par nos politiciens dépendront de la qualité de ces données et devront leur permettre d'anticiper

l'arrivée de problèmes majeurs qui pourraient bloquer l'évolution économique et sociale du pays.

5.5 LA QUALITÉ DES DONNÉES AU SEIN DES ORGANISMES DE LA SÉCURITÉ SOCIALE

« *La non-qualité a un impact d'autant plus important que le système d'information concerné est un instrument d'action sur le réel. Ainsi, dans le domaine de la sécurité sociale belge, plus de 35 milliards d'euros sont annuellement prélevés et redistribués : la qualité des bases de données correspondantes est donc cruciale* » [Bontemps, 2007]

Sur base de la typologie des problèmes de qualité, définie dans le chapitre 3, nous analyserons les données de plusieurs organismes de sécurité sociale afin d'identifier dans les sources de données, des exemples propres à chaque type.

Ensuite, pour chacune des anomalies, nous identifierons la manière de la détecter, puis de la résoudre en utilisant le cycle d'amélioration de la qualité des données. Pour arriver à nos fins, nous utiliserons des instructions SQL testées sous le système de gestion de base de données Oracle.

5.5.1 *MANQUE DE QUALITÉ DANS UNE SOURCE DE DONNÉES*

Les exemples qui vont suivre sont basés sur des cas réels ou vraisemblables mais les données ont été anonymisées afin de protéger les informations à caractères confidentielles.

1) LES PROBLÈMES LIÉS AUX SCHÉMAS

a. Au niveau d'un attribut

- *Les valeurs illégales*
  - ➢ Illustration du problème :

  Ci-après, la table des « périodes trimestrielles » qui est alimentée par un traitement « *batch* ». En rouge, un exemple d'erreur possible.

Matricule	Quarter	Cotisation	Date	DelaidPay
9951110	19924	10000	01/01/1995	951301
8851115	19884	501114	05/02/1989	890131

Description des colonnes :
- Matricule : Identifiant d'un employeur (type : Number).
- Quarter : Trimestre concerné (type : Number, format : YYYYQ)
- Cotisation : Montant en cotisation (type : Number)
- DateEffet : Date d'effet (type : Date)
- DelaidPay : Date d'échéance pour les paiements en cotisation (type : Number, format : CYYMMDD avec C=1 si année >= 2000)

Dans le premier enregistrement, on retrouve une échéance de paiement au *01/13/1995*. Or, le 13ème mois de l'année n'existe pas. Il y a donc bien une erreur. Un meilleur contrôle au niveau du système applicatif, ou un typage correct du champ destiné à stocker l'information, aurait permis de détecter l'erreur avant son enregistrement dans la base de données.

➢ Détection et résolution de l'anomalie :

Dans la définition des métadonnées, le champ « *DelaidPay* » représente une information métier de type « *Date* ». Or, dans la table, l'attribut est défini comme « *Number* ».

Après l'exécution d'un test de « Consistance » via un outil de « *Data Profiling* », la valeur '*951301*' sera déclarée « valeur erronée ».

Pour corriger cette anomalie, il est nécessaire qu'un expert du domaine soit sollicité afin de déterminer la valeur de remplacement, telle que '01/03/1995' ou '13/01/1995'. Dès la valeur de remplacement connue, nous pourrons mettre à jour la valeur fautive via le code SQL suivant :

```
/*Mettre à jour la valeur erronée*/
UPDATE PeriodTrim
SET DelaidPay=TO_NUMBER(TO_DATE('13/01/1995','DD/MM/YYYY'),
'DD/MM/YYYY')
WHERE DelaidPay='951301';

COMMIT;
```

Pour éviter de rencontrer à nouveau ce problème, plusieurs solutions peuvent être appliquées. Citons, par exemple, la standardisation des dates via un attribut de type « *Date* », la mise en place d'un déclencheur (trigger) permettant la validation des données en entrée, ou encore la modification de l'application de manière à ce qu'elle permette uniquement l'enregistrement de données valides.

Nous avons choisi la standardisation, et remplaçons l'attribut existant par un nouvel attribut de type « *Date* ».

```
/*Renommer la colonne erronée*/
ALTER TABLE PeriodTrim
RENAME COLUMN DelaidPay TO DelaidPay_OLD;

/*Ajouter une nouvelle colonne de type Date*/
ALTER TABLE PeriodTrim
ADD DelaidPay DATE;

/*Mettre à jour la nouvelle colonne à partir de l'ancienne*/
UPDATE PeriodTrim
SET DelaidPay=TO_DATE(DelaidPay_OLD+19000000,'YYYYMMDD');

COMMIT;

/*Supprimer la colonne erronée*/
ALTER TABLE PeriodTrim
DROP COLUMN DelaidPay_OLD;
```

Deux stratégies s'offrent à nous afin de garantir une compatibilité de la nouvelle structure de la table avec les applications en entrée. La première consiste à faire illusion de manière à ce que les applications ne détectent aucune différence. La deuxième stratégie nécessite une modification des applications de sorte à s'adapter au nouvel attribut. Choisissons la première stratégie qui permet la transparence pour les applications. L'illusion peut être assurée par la combinaison d'une vue et de deux déclencheurs, tels que représentés, ci-dessous :

```
/*Renommer la table*/
ALTER TABLE PeriodTrim
RENAME TO PeriodTrim_New;
```

```
/*Ajouter une vue reprenant l'ancien format de la table*/
CREATE OR REPLACE VIEW PeriodTrim
AS
SELECT
   Matricule
 , Quarter
 , Period
 , Cotisation
 , DateEffet
 , TO_NUMBER(TO_CHAR(DelaidPay,'YYYYMMDD'))-19000000 as
DelaidPay
FROM PeriodTrim_New;

/*Créer un déclencheur pour les insertions dans la table*/
CREATE OR REPLACE TRIGGER tr_Ins_PeriodTrim INSTEAD OF INSERT ON
PeriodTrim FOR EACH ROW
BEGIN
  INSERT INTO PeriodTrim_NEW
  (Matricule,Quarter,Period,Cotisation,DateEffet,DelaidPay)
  VALUES
(:new.Matricule,:new.Quarter,:new.Period,:new.Cotisation,
:new.DateEffet,to_date(:new.DelaidPay+19000000,'YYYYMMDD'));
END;

/*Créer un déclencheur pour les mises à jour dans la table*/
CREATE OR REPLACE TRIGGER tr_Upd_PeriodTrim INSTEAD OF UPDATE ON
PeriodTrim FOR EACH ROW
BEGIN
  UPDATE PeriodTrim_New
  SET DelaidPay=TO_DATE(:new.DelaidPay+19000000,'YYYYMMDD')
  WHERE Matricule=:old.Matricule
  AND Quarter=:old.Quarter
  AND Period=:old.Period;
END;
```

b. Au niveau d'une table

- *La violation d'unicité*

  ➢ Illustration du problème :

Ci-après, un exemple basé sur la table des « assurés sociaux » (employés, ouvriers,…), qui a pour objectif d'enregistrer les informations concernant tous les « assurés sociaux » référencés dans une déclaration multifonctionnelle envoyée par un employeur.

En rouge, un exemple de violation d'unicité rencontré lors de l'enregistrement d'un nouvel assuré social.

Table des assurés sociaux

ID	Nom	Prénom	RegNat
100	Dupont	Roger	123456-123-12
101	Dupont	Roger	123456-123-12

Description des colonnes :
- ID : Identifiant de l'assuré social (type : Number).
- Nom: Nom de l'assuré social (type : Char).
- Prénom: Prénom de l'assuré social (type : Char).
- RegNat: Numéro de Registre National de l'assuré social (type : Char).

Dans ce cas de figure, nous avons deux enregistrements pour une seule et même personne. Nous avons donc bel et bien une redondance d'information. A l'origine de la duplication, un employeur a entré sa déclaration multifonctionnelle le 01/01/2007 à 8h 50min 10sec. Lorsqu'un employeur entre une déclaration, le système vérifie si les employés référencés dans la déclaration sont connus de la table des « assurés sociaux ». Dans le cas contraire, un nouvel enregistrement est créé dans la table avec un nouvel identifiant. Cette vérification prend une dizaine de seconde. Dans ce cas-ci, un nouvel employé a été détecté, il s'agit de Mr '*Roger Dupont*'. Le problème survient lorsque trois secondes plus tard, l'employeur envoi une seconde fois sa déclaration. A ce moment-là, lorsque le système fait la vérification, l'assuré social n'existe pas encore, et un nouvel enregistrement est alors généré. Ce problème relève de ce qu'on appelle « la gestion des accès concurrents » dont nous ne feront pas état dans ce mémoire.

➢ Détection et résolution de l'anomalie :

Dans les métadonnées sur la table des « Assurés Sociaux », l'information d'unicité sur la colonne 'RegNat' permettra à l'outil de « *Data Profiling* » de détecter le doublon. En l'absence de cette information, comme nous l'avons vu dans le chapitre précédent, le « *Data Matching* » permet de détecter le doublon.

Ci-dessous, un exemple de code permettant la détection des doublons en fournissant l'identifiant de l'enregistrement dédoublé ainsi que l'identifiant du doublon à supprimer :

```
/*Détection des doublons*/
WITH
  T_Source as
      (SELECT ID as ID_Source,RegNat
       FROM AssurSoc T1
       WHERE EXISTS(
          SELECT 1 FROM AssurSoc T2
          WHERE T1.ID < T2.ID
            AND T1.RegNat = T2.RegNat)),
  T_Doublon as
      (SELECT ID as ID_Doublon,RegNat
       FROM AssurSoc T1
       WHERE EXISTS(
          SELECT 1 FROM AssurSoc T2
          WHERE T1.ID > T2.ID
            AND T1.RegNat = T2.RegNat))
SELECT ID_Source,ID_Doublon,T_Source.RegNat
FROM T_Source JOIN T_Doublon
  ON T_Source.RegNat = T_Doublon.RegNat
 AND T_Source.ID_Source <> T_Doublon.ID_Doublon;
```

La résolution du problème se fera en deux étapes. Tout d'abord, il sera nécessaire de mettre à jour l'ensemble des tables comprenant des clés étrangères référençant la clé primaire de l'enregistrement dédoublé. Ensuite, l'enregistrement dédoublé pourra être supprimé. Ci-dessous, les deux étapes :

```
/*Mise à jour des clés étrangères*/
UPDATE <Table>
SET AssurSoc_FK =100
WHERE AssurSoc_FK =101;

COMMIT;

/*Suppression de l'enregistrement dédoublé*/
DELETE FROM AssurSoc
WHERE ID=101;

COMMIT;
```

Afin d'éviter que ce type de problème se présente à nouveau, on pourra ajouter une contrainte d'unicité sur la colonne 'RegNat' :

```
/*Ajout d'une contrainte d'unicité*/
ALTER TABLE AssurSoc
ADD CONSTRAINT RegNat_Unique UNIQUE (RegNat);
```

2) LES PROBLÈMES LIÉS AUX INSTANCES

a. Au niveau d'un attribut

- *Les erreurs de mots mal orthographiés*

  ➢ Illustration du problème :

  Ci-dessous, la table des « délais de paiement des cotisations trimestrielles » qui est alimentée manuellement de manière périodique.

  En rouge, un exemple d'erreur de date mal orthographié :

Table des « délais de paiement des cotisations trimestrielles »

Quarter	Pay_LD	Pay_LDT	Pay_SSD
19873	31/10/1987	04/11/1987	15/12/2087
19884	31/01/1989	04/02/1989	15/03/1989

Description des colonnes :
- Quarter : Trimestre concerné (type : Number, format : YYYYQ)
- Pay_LD : Date d'échéance légale de paiement pour les employeurs (type : Date)
- Pay_LDT : Date d'échéance légale de paiement pour les employeurs + jours de tolérance (type : Date)
- Pay_SSD : Date d'échéance légale de paiement pour les secrétariats sociaux (type : Date)

Dans le premier enregistrement concernant le troisième trimestre 1987, la date d'échéance légale de paiement des cotisations trimestrielles pour les secrétariats sociaux est indiquée comme étant le *15/12/2087* au lieu du *15/12/1987*.

➢ Détection et résolution de l'anomalie :

Dans les métadonnées, l'ajout d'une règle du genre : « l'année de [Pay_SSD] doit être comprise entre l'année de [Quarter] et l'année de [Quarter]+1 » permettra au « *Data Profiling* » de détecter l'erreur.

Une mise à jour du champ sera nécessaire pour corriger l'anomalie :

```
/*Mise à jour de l'anomalie*/
UPDATE  Dcl_Pay
SET Pay_SSD =TO_DATE('15/12/1987','DD/MM/YYYY')
WHERE Pay_SSD =TO_DATE('15/12/2087','DD/MM/YYYY');

COMMIT;
```

Afin d'éviter l'apparition de nouvelles anomalies de ce type, un mécanisme de détection instantanée peut être mis en place. Ce dernier va vérifier que « l'année de [Pay_SSD] est comprise entre l'année de [Quarter] et l'année de [Quarter]+1 ». Ci-dessous, un exemple de code SQL également adapté pour gérer les même type d'erreurs des attributs [Pay_LD] et [Pay_LDT] :

```
/*Contrôler la validité des dates en entrée*/
CREATE OR REPLACE TRIGGER tr_InsUpd_Dcl_Pay BEFORE INSERT OR
UPDATE ON Dcl_Pay FOR EACH ROW
BEGIN
  IF NOT(TO_CHAR(:new.PAY_LD,'YYYY') BETWEEN
SUBSTR(:new.Quarter,1,4) AND SUBSTR(:new.Quarter,1,4)+1) THEN
     raise_application_error (-20001,'La date d''échéance légale
de paiement pour les employeurs est en dehors des limites.');
  END IF;
  IF NOT(TO_CHAR(:new.Pay_LDT,'YYYY') BETWEEN
SUBSTR(:new.Quarter,1,4) AND SUBSTR(:new.Quarter,1,4)+1) THEN
     raise_application_error (-20002,'La date de tolérance
légale de paiement pour les employeurs est en dehors des
limites.');
  END IF;
  IF NOT(TO_CHAR(:new.Pay_SSD,'YYYY') BETWEEN
SUBSTR(:new.Quarter,1,4) AND SUBSTR(:new.Quarter,1,4)+1) THEN
     raise_application_error (-20003,'La date d''échéance légale
de paiement pour les secrétariats sociaux est en dehors des
limites.');
  END IF;
END;
```

Le déclencheur, ci-dessus, permettra de détecter les nouvelles anomalies et d'avertir l'utilisateur sur l'impossibilité d'écrire l'éventuelle information erronée. Il sera alors invité à encoder une nouvelle date.

- *Les abréviations*

  ➢ Illustration du problème :

  L'adresse est un bon exemple où les abréviations sont souvent sources de surcharge de travail et de confusion dans les sources de données.

  Ci-dessous, un exemple de problème de doublon engendré par l'utilisation d'abréviations dans un attribut de la table des « données signalétiques de l'employeur ».

Table des « données signalétiques de l'employeur »

Matr	Denom	Adress	Cp	City	DateMaj
1994444	KREB SA	R DE L EGLISE 10	1000	BRUXELLES	15/12/1997
1994444	KREB SA	RUE DE L EGLISE 10	1000	BRUXELLES	10/05/1998

Description des colonnes :
- Matr : Matricule, identifiant de l'employeur (type : Char)
- Denom : Dénomination de l'employeur (type : Char)
- Adress : Adresse de l'employeur (type : Char)
- Cp : Code postal (type : Number)
- City : Ville (type : Char)
- DateMaj : Date de la modification (type : Date)

On peut lire dans l'exemple que la société '*KREB SA*' a changé d'adresse le *10/05/1998*, alors que visiblement l'adresse est la même.

➢ Détection et résolution de l'anomalie :

Grâce à l'utilisation de données géographiques en entrée, le processus de « *Data Matching* » pourra retrouver l'adresse correcte à utiliser. La mise à jour pourra, dès lors, être faite via un simple « *Update* » :

```
/*Mise à jour de l'anomalie*/
UPDATE  Empl
SET Adress='RUE DE L EGLISE 10'
WHERE Adress='R DE L EGLISE 10';

COMMIT;
```

Ci-dessous, l'état de la table après transformation de l'abréviation :

Matr	Denom	Adress	Cp	City	DateMaj
1994444	KREB SA	RUE DE L EGLISE 10	1000	BRUXELLES	15/12/1997
1994444	KREB SA	RUE DE L EGLISE 10	1000	BRUXELLES	10/05/1998

Après correction de l'adresse, le processus de « *Data Matching* » va signaler la présence d'un doublon dans la table. Il ne restera alors plus qu'à supprimer ce doublon. Ci-dessous, une procédure SQL pour la détection et la suppression du doublon.

```
/*Détection des doublons*/
SELECT T1.*
FROM
Empl T1
JOIN Empl T2 ON T1.Matr=T2.Matr
Where
    T1.Adress=T2.Adress
AND T1.DateMaj > T2.DateMaj
AND NOT EXISTS
(SELECT 1 FROM Empl T3
 WHERE
       T3.Matr = T1.Matr
   AND T3.DateMaj < T1.DateMaj
   AND T3.DateMaj > T2.DateMaj);


/*Suppression des doublons*/
DELETE FROM Empl
WHERE
ROWID in
(
SELECT T1.ROWID
FROM
Empl T1
JOIN Empl T2 ON T1.Matr=T2.Matr
Where
     T1.Adress=T2.Adress
AND  T1.DateMaj > T2.DateMaj
AND  NOT EXISTS
(SELECT 1 FROM Empl T3
 WHERE
       T3.Matr  = T1.Matr
   AND T3.DateMaj < T1.DateMaj
   AND T3.DateMaj > T2.DateMaj)
);
```

```
COMMIT;
```

- *Les valeurs mal attribuées*

  ➢ Illustration du problème :

  Examinons un exemple de valeurs mal attribuées qui se base, une fois encore, sur la table des « données signalétiques de l'employeur ».

Table des « données signalétiques de l'employeur »

Matr	Denom	Adress	Cp	City
1993333	CORA	RUE DE LA BOTTE 40	0	75008 PARIS
1992222	Monsieur MENUISERIE	RUE DU MARAIS 30	1000	BRUXELLES

Description des colonnes :
- Matr : Matricule, identifiant de l'employeur (type : Char)
- Denom : Dénomination de l'employeur (type : Char)
- Adress : Adresse de l'employeur (type : Char)
- Cp : Code postal (type : Number)
- City : Ville (type : Char)

Dans le premier enregistrement, le code postal a été encodé au mauvais endroit. En effet, le champ « *City* » contient le code postal ainsi que le nom de la ville. De plus, dans le code postal, nous trouvons la valeur '*0*' alors que cette valeur n'est pas permise pour un code postal.

➢ Détection et résolution de l'anomalie :

Dans les métadonnées, l'attribut « Cp » est défini comme un champ numérique devant contenir des codes postaux du monde entier. Ajoutons dans les métadonnées que « Cp » doit contenir un entier positif différent de zéro. Précisons également que l'attribut « *City* » ne peut contenir que du texte.

Après avoir précisé les métadonnées, nous pourrons lancer le traitement de « *Data Profiling* » qui nous indiquera avoir trouvé les deux anomalies attribuées au matricule '*1993333*'.

Dès les anomalies détectées, nous pouvons procéder à la correction. Après une analyse rapide, nous nous apercevons que le code postal est inclus dans l'attribut « *City* ». Ci-dessous, le code de mise à jour de l'enregistrement :

```
/*Mise à jour de l'anomalie*/
UPDATE  Empl
SET Cp=75008
   ,City='PARIS'
WHERE
     Matr=' 1993333'
 AND City='75008 PARIS';

COMMIT;
```

Nous pouvons éviter que ces erreurs se présentent à nouveau en ajoutant une contrainte et un déclencheur sur la table. Ci-dessous, le code à ajouter :

```
/*Le code postal doit être positif*/
ALTER TABLE Empl
ADD CONSTRAINT check_Code_Postal
    CHECK (Cp > 0);

/*Vérifier que City ne contient que du texte*/
CREATE OR REPLACE TRIGGER tr_InsUpd_Empl BEFORE INSERT OR UPDATE
OF City ON Empl FOR EACH ROW
BEGIN
   IF NOT(owa_pattern.match(:new.City, '^\D*$')) THEN
      raise_application_error (-20001,'La ville ne peut pas
contenir de valeur numérique.');
   END IF;
END;
```

b. Au niveau d'un enregistrement

- *La violation des dépendances fonctionnelles*

  ➢ Illustration du problème :

Ci-dessous, un exemple basé sur la table des « Personnes de contact » qui a pour vocation d'enregistrer les renseignements des personnes responsables pour un ensemble défini d'employeurs. Ces personnes sont réparties dans différents services de l'organisme. Certaines valeurs d'un enregistrement dépendent du type de service auquel appartient la personne (cf. Description des colonnes). Ajoutons également que cette table est alimentée manuellement.

Ci-dessous, un exemple d'enregistrement pour lequel une violation de dépendance fonctionnelle a été détectée.

Table « Personnes de contact »

Dienst	Adm	BeginGrens	EndGrens	PersID	Pers	PersTel
C	E	0	999	4411	Durand Pierre	9999

Description des colonnes :
- Dienst : Service du responsable (type : Char). Les valeurs possibles sont ' J ' pour le service Recouvrement Judiciaire, ' C ' pour le service Perception et ' I ' pour le service Identification.
- Adm : Situation administrative de l'employeur (type : Char).
  Valeurs permises :
  Case Dienst
  C : {0,1,2,3}
  I : {*}
  J : {A,B,C,D,E}
  End Case
- BeginGrens : Limite inférieure de l'ensemble des numéros de matricule de l'employeur à gérer par le responsable (type : Num).
- EndGrens : Limite supérieure de l'ensemble des numéros de matricule de l'employeur à gérer par le responsable (type : Num).
- PersID : Identifiant du responsable (type : Num).
- Pers : Nom du responsable (type : Char).
- PersTel : Numéro de téléphone du responsable (type : Char).

En analysant la description des colonnes de la table, nous voyons que la situation administrative de l'employeur (=colonne « Adm ») dépend du service concerné. Dans notre exemple, nous avons comme valeurs '*C*' pour le service et '*E*' pour la situation administrative. Pourtant, ces deux valeurs ne peuvent exister dans un même enregistrement.

➢ Détection et résolution de l'anomalie :

Dans les métadonnées, les règles de dépendances fonctionnelles entre « Dienst » et « Adm » sont clairement énoncées. Le processus de « *Data Profiling* » n'aura, par conséquent, aucun mal à trouver l'enregistrement erroné.

Après enquête auprès des responsables métier, il s'est avéré qu'il s'agissait d'un enregistrement concernant le service « Recouvrement Judiciaire », et que la valeur du champ « Dienst » devait être ' J ' et non ' C '.

La résolution du problème passe, par conséquent, par l'exécution d'une mise à jour du champ « Dienst ». Ci-dessous, le code SQL exécuté :

```
/*Mise à jour de l'anomalie*/
UPDATE PersContact
SET Dienst='J'
WHERE
    PersID=4411
AND Dienst='C'
AND Adm='E';

COMMIT;
```

Pour résoudre définitivement les problèmes de ce type, il est nécessaire d'ajouter une contrainte sur la table :

```
ALTER TABLE PersContact
ADD CONSTRAINT check_Dienst_Adm
   CHECK ((Dienst='C' AND Adm in('0','1','2','3'))
       OR (Dienst='J' AND Adm in('A','B','C','D','E'))
       OR (Dienst='I'));
```

c. Au niveau d'une table

- *Les informations redondantes*

  ➢ Illustration du problème :

Examinons un exemple d'information redondante. Il se base sur la table des « Anomalies DMFA » qui a pour objectif d'enregistrer les renseignements sur les anomalies détectées dans une déclaration DMFA[3] introduite par un employeur.

[3] DMFA est l'abréviation de « Déclaration Multifonctionnelle » et « MultiFonctionele Aangifte ». Il s'agit de la dénomination donnée aux déclarations trimestrielles que tout employeur doit envoyer à l'Onss.

Table des « Anomalies DMFA »

EDPid	VersionNb	ErrorNb	BlokNb	Status	Date
A10501	1	119	15	Detected	14/11/2007
A10501	1	54	28	Detected	14/11/2007
A10501	2	119	15	Detected	24/11/2007
A10501	2	54	28	Closed	08/12/2007
A10501	3	119	15	Closed	08/12/2007

Description des colonnes :
- EDPid : Identifiant de la DMFA (type : Char)
- VersionNb : Version de la DMFA (type : Number)
- ErrorNb : Numéro de l'erreur (type : Number)
- BlokNb : Numéro du bloc concerné (type : Number)
- Status : Statut de l'anomalie (type : Char)
- Date : Date de la mise à jour (type : Date)

La table « Anomalies DMFA » est alimenté de la façon suivante : chaque fois qu'une version de la déclaration est envoyée par l'employeur, celle-ci arrive dans le système opérationnel. Un contrôle minutieux est, alors, lancé afin de détecter les anomalies dans cette nouvelle version. Lors de la détection d'une anomalie, le système ajoutera un enregistrement dans la table « Anomalies DMFA » uniquement si l'anomalie concernée ne s'y trouve pas encore représentée. Pour les anomalies déjà répertoriées dans la table, le système indique, par une mise à jour du statut de '*Detected*' vers '*Closed*', si elles sont résolues.

Dans le cas qui nous occupe, le problème est le suivant : au *24/11/2007*, une anomalie (ErrorNb=*119* et BlockNb=*15*) a été ajoutée dans la table alors qu'elle y était déjà référencée dans la première version de la DMFA (version=*1*). Par la suite, le *08/12/2007*, une version '*3*' de la DMFA corrigeait cette anomalie. Malheureusement, l'anomalie détectée dans la version 1 persiste sous le statut '*Detected*'.

➢ Détection et résolution de l'anomalie :

Dans les métadonnées, il faut s'assurer que les attributs <EDPid, ErrorNb, BlockNb, Status> soient bien indiqués comme uniques.

Si c'est le cas, la procédure de « *Data Profiling* » va détecter l'erreur mentionnée ci-dessus.

L'erreur pourra être corrigée en supprimant l'enregistrement dupliqué. Le code, ci-dessous, permet d'assurer cette tâche :

```
/*Suppression de l'erreur*/
DELETE FROM AnoDmfa
WHERE EDPid='A10501'
  AND ErrorNb=119
  AND BlockNb=15
  AND Status='Detected';
```

S'il n'est pas prévu qu'une anomalie déjà corrigée puisse réapparaître comme '*Detected*' dans une version ultérieure, la résolution de ce problème à long terme est l'ajout d'une contrainte d'unicité sur les quatre champs :

```
ALTER TABLE AnoDmfa
ADD CONSTRAINT check_Unique_AnoDmfa
   UNIQUE (EDPid,ErrorNb,BlockNb,Status);
```

- *Les enregistrements contradictoires*

➢ Illustration du problème :

Voici un exemple basé sur deux tables : la table « Employer Declaration » qui permet d'enregistrer les informations sur une déclaration DMFA, et la table des « Anomalies DMFA » qui enregistre les anomalies détectées sur une déclaration.

Ci-dessous, un exemple d'enregistrements contradictoires rencontrés entre ces deux tables :

1) Table « Employer Declaration »

EDPid	VersionNb	UserID	UserQuality
10000	1	XRE	10

2) Table des « Anomalies DMFA »

EDPid	VersionNb	UserID	UserQuality	AnoClass
10000	1	JRR	50	13

Description des colonnes :
- EDPid : Identifiant permanent de la déclaration (type : Number).
- VersionNb: Numéro de version de la déclaration (type : Number).
- UserID : Identifiant du déclarant de la déclaration (type : Char).
- UserQuality : Qualité du déclarant (type : Number).
- AnoClass : Classe de l'anomalie (type : Char).

La première table « Employeur Declaration » contient le détail d'une version de la déclaration DMFA d'un employeur. La seconde table « Anomalies DMFA » contient les anomalies détectées sur une version de la déclaration.

Dans la table « Employer Declaration », on voit qu'une personne, avec le « UserID » '*XRE*' et le « UserQuality » de type '*10*' (=employeur), a introduit une déclaration. Il s'agit de la première version de cette déclaration qui a reçu l'identifiant '*10000*'. Ensuite, une anomalie a été détectée sur cette déclaration. Ainsi, un enregistrement a été créé dans la table « Anomalies DMFA ». Nous savons que la personne qui génère une anomalie est la même que celle qui entre la déclaration dans le système. Par conséquent, le champ « UserId » devrait être le même pour les deux enregistrements concernés. Or, nous avons dans la deuxième table une personne ayant le « UserId » '*JRR*' et non '*XRE*'. Nous avons, donc, bien une contradiction dans l'information fournie par ces deux tables.

➢ Détection et résolution de l'anomalie :

Comme il s'agit d'une incohérence entre deux tables différentes, nous n'utiliserons pas le « *Data Profiling* » pour détecter l'erreur mais, plutôt, le « *Data Matching* » qui pourra s'acquitter de cette tâche. Après validation avec les analystes métier, les valeurs incorrectes pourront être mises à jour de la manière suivante :

```
/*Correction de l'erreur*/
UPDATE AnoDmfa
Set (UserID,UserQuality)=(SELECT DISTINCT UserID,UserQuality
                          FROM EmplDeclaration
                          WHERE EDPID=10000
                            AND VersionNb=1)
WHERE EDPID=10000
  AND VersionNb=1;

COMMIT;
```

Pour résoudre les éventuelles autres erreurs, une mise à jour complète de la table pourrait être réalisée. Le code SQL à exécuter :

```
/*Suppression des erreurs*/
UPDATE AnoDmfa T1
Set (UserID,UserQuality)=(SELECT DISTINCT UserID,UserQuality
                          FROM EmplDeclaration T2
                          WHERE T1.EDPID=T2.EDPID
                            AND T1.VersionNb=T2.VersionNb)
WHERE
EXISTS
(SELECT 1 FROM EmplDeclaration T3
 WHERE T1.EDPID=T3.EDPID
   AND T1.VersionNb=T3.VersionNb
   AND (   T1.UserID <> T3.UserID
        OR T1.UserQuality <> T3.UserQuality));

COMMIT;
```

Afin d'éviter de nouvelles erreurs de ce type, un déclencheur peut être mis en place. Ci-dessous, son code SQL :

```
CREATE OR REPLACE TRIGGER tr_InsUpd_AnoDmfa
BEFORE INSERT OR UPDATE OF UserID,UserQuality ON AnoDmfa FOR
EACH ROW
DECLARE
  v_UserID Varchar2(20);
  v_UserQuality NUMBER;
BEGIN
  SELECT DISTINCT UserID,UserQuality into v_UserID,v_UserQuality
  FROM EmplDeclaration
  WHERE EDPID=:new.EDPID
    AND VersionNb=:new.VersionNb;

  IF (:new.UserID <> v_UserID
      OR :new.UserQuality <> v_UserQuality) THEN
     raise_application_error (-20001,'L''identifiant ainsi que
la qualité du déclarant doivent être identique aux valeurs
originales (voir table "EmplDeclaration").');
  END IF;
END;
```

- *Les identifiants significatifs instables*

  ➢ Illustration du problème :

Le « numéro de matricule » d'un employeur est un bon exemple de clé significative instable. Ce numéro est utilisé comme identifiant d'un employeur dans la plupart des applications.

En effet, le « numéro de matricule » d'un employeur est une clé « significative » car il est lié à au moins une caractéristique de l'objet qu'il représente, dans ce cas-ci, la langue de l'employeur.

La répartition se fait comme suit :

Les matricules néerlandophones de :	100.000	à 499.999
	900.000	à 999.999
	1.100.000	à 1.499.999
Les matricules francophones de :	500.000	à 899.999
	1.500.000	à 1.999.999

Un problème survient lorsqu'un employeur change de langue, par exemple, si l'employeur déménage de la Wallonie vers la Flandre, ou inversement. Ce changement engendre une modification de l'identifiant de l'employeur car ce dernier reçoit un nouveau « numéro de matricule ». Ce changement d'identifiant engendre d'énormes difficultés de gestion de l'information car pour pouvoir assurer le suivi de l'employeur, il est indispensable de pouvoir faire le lien entre l'ancien et le nouveau matricule.

Ajoutons qu'il existe une table, appelée table des « transferts d'employeur », qui contient les correspondances entre l'ancien numéro d'employeur et le nouveau. Le problème est qu'un fonctionnaire peut, de temps à autre, être confronté aux données d'un employeur ayant déménagé 3 ou 4 fois son siège social d'une région à l'autre. La situation peut alors rapidement devenir complexe à analyser, comme par exemple, calculer la somme des dettes des employeurs.

Ci-dessous, un exemple qui se base sur la table des « données signalétiques de l'employeur » ainsi que sur la table des « transferts d'employeur ».

Table des « données signalétiques de l'employeur »

Matr	Denom	Adress	Cp	City	Statut
1510000	MACOL	RUE DE LA BOTTE 40	7000	Mons	Transfert
1110000	MACOL	BRUGGESTRAAT 5	1500	Halle	Transfert
1710000	MACOL	RUE DU BOIS 15	7850	Enghien	Transfert
1493333	MACOL	REDENSTRAAT 2	1500	Halle	Actif

Description des colonnes :
- Matr : Matricule, identifiant de l'employeur (type : Char)
- Denom : Dénomination de l'employeur (type : Char)
- Adress : Adresse de l'employeur (type : Char)
- Cp : Code postal (type : Number)
- City : Ville (type : Char)
- Statut : Statut du numéro d'employeur (type : Char)

Table des « transfert d'employeur »

	Matr_From	Matr_To	DateMaj
→	1510000	1110000	10/05/2005
	550005	1486662	11/05/2005
→	1110000	1710000	10/06/2005
→	1710000	1493333	10/12/2005

Description des colonnes :
- Matr_From : Matricule source (type : Char)
- Matr_To : Matricule destination (type : Char)
- DateMaj : Date de mise à jour (type : Date)

L'exemple représente une société, dénommée « MACOL », qui a déménagé trois fois dans une période d'un an en alternant commune francophone puis néerlandophone. Ce qui a eu pour conséquence la création de quatre matricules différents pour le même employeur.

➢ Détection et résolution de l'anomalie :

Les problèmes engendrés par la présence de plusieurs identifiants pour un même employeur ont un impact sur l'interprétation des utilisateurs de l'information, et ne représentent en rien des erreurs formelles. Il n'est donc pas possible de détecter ces problèmes de conception via les outils présentés dans le cycle d'amélioration de la qualité des données.

La solution à ce problème de conception est de mettre en place un nouvel identifiant stable comme, par exemple, une « clé technique » (« *surrogate key* »).

d. Au niveau de la base de données

- *La mauvaise gestion des dates de transaction*

  ➢ Illustration du problème :

  Ci-dessous, un exemple basé sur la table des « Personnes de contact ». Comme expliqué dans un cas précédent (cf. « la violation des dépendances fonctionnelles »), cette table a pour objectif d'identifier les personnes responsables pour un ensemble déterminé d'employeurs. Elle est alimentée manuellement, et aucun champ n'a été prévu pour contenir une date de transaction.

  Pour rappel, une date de transaction est associée à un ou plusieurs champs d'un enregistrement, et contient la date de la dernière mise à jour des attributs auxquels elle est associée.

  Vous trouverez, ci-dessous, un exemple d'enregistrement de cette table.

Table des « Personnes de contact »

Dienst	Adm	BeginGrens	EndGrens	PersID	Pers	PersTel
J	E	0	999	4411	Durand Pierre	9999

Description des colonnes :
- Dienst : Service du responsable (type : Char). Les valeurs possibles sont ' J ' pour le service Recouvrement Judiciaire, ' C ' pour le service Perception et ' I ' pour le service Identification.
- Adm : Situation administrative de l'employeur (type : Char).
  Valeurs permises :

```
Case Dienst
    C : {0,1,2,3}
    I : {*}
    J : {A,B,C,D,E}
End Case
```

- BeginGrens : Limite inférieure de l'ensemble des numéros de matricule de l'employeur à gérer par le responsable (type : Num).
- EndGrens : Limite supérieure de l'ensemble des numéros de matricule de l'employeur à gérer par le responsable (type : Num).
- PersID : Identifiant du responsable (type : Num).
- Pers : Nom du responsable (type : Char).
- PersTel : Numéro de téléphone du responsable (type : Char).

Comme nous ne pouvons pas définir la date de modification des attributs de la table, il est, par exemple, impossible d'identifier depuis quand un responsable s'occupe d'une liste d'employeur.

➢ Détection et résolution de l'anomalie :

Aucun des outils présentés dans le cycle d'amélioration de la qualité ne permet de détecter ce type de problème. En effet, l'information est manquante, et aucune zone ne permet de l'ajouter.

Afin de résoudre ce manquement, nous allons ajouter à la table une nouvelle colonne pour la date de transaction. Son alimentation se fera par l'intermédiaire d'un déclencheur. Ci-dessous, le code SQL permettant de remplir ces objectifs :

```
/*Ajout de la colonne DateMaj*/
ALTER TABLE PersContact
ADD DateMaj DATE;

/*Ajout d'un déclencheur pour remplir la date de transaction*/
CREATE OR REPLACE TRIGGER tr_InsUpd_PersContact_DateMaj
BEFORE INSERT OR UPDATE ON PersContact FOR EACH ROW
BEGIN
  :new.DateMaj:=SYSDATE;
END;
```

5.5.2 *MANQUE DE QUALITÉ APRÈS L'INTÉGRATION DE PLUSIEURS SOURCES DE DONNÉES*

1) LES PROBLÈMES LIÉS AUX INSTANCES

- *Les différences de représentation et d'interprétation*

➢ Illustration du problème :

Au sein des différents systèmes d'information de l'entreprise, l'identifiant pour un employeur est le « Matricule ». Dans la base de données dédiée à la « déclaration DMFA », le numéro de matricule est composé de 7 caractères suivant le format « 1234567 » ; alors que dans la base de données « Répertoire des employeurs », le « Matricule » est représenté suivant le format « 5671234 », c'est-à-dire que les trois derniers chiffres sont placés devant les quatre premiers.

Table des « Employeurs » (BD Répertoire des employeurs)

Matr	Denom
1994444	I. G. HABITATIONS SPRL

Table « Emloyer Declaration » (BD Déclaration DMFA)

EDPid	Matr	VersionNb	UserID	UserQuality
10000	4444199	1	XRE	10

On observe dans cet exemple deux représentations différentes pour un seul et même matricule. De plus, nous remarquons qu'il s'agit ici d'une différence de représentation concernant un identifiant.

- Détection et résolution de l'anomalie :

La détection des différences de représentation dans les bases de données ne pourra pas être assurée par le processus d'amélioration de la qualité.

Néanmoins, le processus de « Standardisation » permettra d'uniformiser les conventions de représentation et d'interprétation au niveau de l'entreprise, et permettra d'éviter les différences de représentation de la même information dans les différentes bases de données de l'entreprise.

5.6 CONCLUSION

Pour les organismes publics, le défi de la qualité de l'information est primordial car, contrairement à une entreprise ordinaire du secteur privé où le manque de qualité de l'information lui fait perdre en efficacité et en gain, dans le secteur public, le droit à l'erreur ne peut, théoriquement, pas être admis. En effet, l'institution publique a le devoir légal de traiter les citoyens de façon équitable. Une erreur, si minime qu'elle soit, pourrait avoir des conséquences non négligeables pour l'entreprise et les citoyens concernés.

Cette étude de cas nous a permis d'illustrer le fait que les organismes de la sécurité sociale ne sont pas à l'abri des problèmes de qualité de l'information. En effet, le manque de qualité dans les données est omniprésent dans le monde de l'entreprise, et touche tous les types de sociétés du secteur privé au secteur public. La plupart des problèmes ont pu être détectés grâce aux outils de « *Data Profiling* » et de « *Data Matching* ». Pour ce qui est des solutions apportées, l'implémentation des contraintes du domaine a permis de résoudre la majorité des problèmes.

Chapitre 6 : CONCLUSION

Dans une première partie de ce mémoire, nous avons défini la qualité des données. Comme nous avons pu l'apprendre, l'appréciation de la qualité des données ne peut jamais être fixée de manière définitive car elle évolue sans cesse en fonction des besoins exprimés par les utilisateurs. Nous avons également parcouru les différentes conséquences que peut provoquer une mauvaise qualité dans les données. La mauvaise qualité impacte aussi bien l'entreprise que les utilisateurs de l'information. On a vu que les erreurs d'encodage ainsi que le manque d'implémentation des contraintes sur les données étaient les premières causes des problèmes de qualité.

Dans une deuxième partie, nous avons parcouru une liste non exhaustive des problèmes de qualité qui peuvent apparaître dans les données. Les problèmes ont été classés suivant deux catégories. Nous avions d'une part les problèmes de qualité rencontrés à l'intérieur d'une base de données. Et d'autre part, nous avions les problèmes de qualité rencontrés lorsqu'on intègre des données à partir de plusieurs sources différentes.

Dans une troisième partie du document, nous avons présenté une méthodologie permettant de résoudre la plupart des problèmes de qualité. Celle-ci se base sur un processus itératif reposant sur quatre étapes. Une première étape, appelée « *Data Profiling* », permet de comparer les données réellement inscrites dans la base avec leurs métadonnées respectives de sorte à pouvoir détecter les incohérences existantes. Il permet de détecter les anomalies tant au niveau des contraintes d'intégrités qu'au niveau du contenu. Chaque itération de ce processus d'amélioration qualité permettra de corriger et de renforcer les métadonnées.
Une deuxième étape du processus d'amélioration de la qualité est appelé « *Data Standardization* » et consiste à modifier les données de sorte que celles-ci suivent les règles métiers et les standards de l'entreprise.
Nous avons, ensuite, défini la troisième étape appelée « *Data Matching* ». Celle-ci a comme objectif la détection des incohérences et des doublons dans les données. Cette détection nécessite l'exécution d'algorithmes complexes et très coûteux en termes de temps processeurs. Pour cela, la procédure de détection des doublons démarre par une première phase appelée « processus de recherche » qui permet de limiter le nombre d'enregistrement à analyser en éliminant les données non compatibles.
Finalement, nous avons appris qu'une dernière étape du cycle d'amélioration de la qualité est appelée « *Data Monitoring* » et permet de suivre l'évolution de la qualité des données

dans l'entreprise par la diffusion de rapports sur l'état des données, les nouvelles anomalies détectées ainsi que les manipulations nécessaires qui ont été accomplies.

Dans la dernière partie du mémoire, une étude de cas, réalisée sur des organismes publics du domaine de la Sécurité Sociale, nous a permis de démontrer l'importance d'utiliser un outil d'amélioration de la qualité des données. Nous avons pu remarquer que les problèmes de qualités étaient nombreux et que la résolution de ceux-ci ne nécessite, souvent, que peu de manipulation. L'étude, nous a permis de découvrir que dans la majorité des cas, l'application des règles simples de construction d'une base de données (clés primaires, clés secondaires, typage correct, clause not null, ...), lors de la création des tables, aurait, dès le début, permis d'éviter un grand nombre de situation problématique.

Ce mémoire nous a permis de détailler les différentes étapes à prendre en considération lors d'un projet d'amélioration de la qualité des données. Comme suite à ce mémoire, il pourrait être intéressant de se pencher sur la mise en place d'un système de gestion de la qualité automatisé. Un système semi-automatisé devra également être pris en compte de manière à permettre, dans certains cas, une validation préalable, par les responsables métier, des corrections proposées. Les problèmes de qualité sont tellement nombreux qu'un traitement purement manuel n'est pas envisageable.
Comme nous avons pu le démontrer dans l'étude de cas, la plupart des problèmes (abréviation, absence de clé primaire,...) ont engendré l'apparition de doublons dans les tables. Nous l'avons vu, un outil de « *Data Matching* » devra être utilisé pour la détection des doublons. Il aurait, également, été intéressant de s'attarder sur les techniques utilisés par ce genre d'outil pour parvenir à ces fins (fuzzy matching, distance de Levenshtein,...).

Ce mémoire nous a permis de découvrir qu'une base de données ne peut pas être modélisée en fonction de la confiance qu'on porte aux futurs utilisateurs, ou en focalisant son attention sur l'économie en terme d'espace disque et de coût processeur. Il est important d'implémenter toutes les règles et contraintes nécessaires afin d'éviter tout écart de qualité. Il sera plus avantageux pour l'entreprise de privilégier la détection des anomalies avant leur enregistrement dans la source de données plutôt que de consacrer du temps et beaucoup d'argent dans leur résolution. De plus, l'anticipation des anomalies permettra également d'éviter la création de préjudices, non fondé, envers les utilisateurs.

Chapitre 6 : Conclusion

Chapitre 7 : BIBLIOGRAPHIE

[1] BCSS, http://www.ksz.fgov.be, (last revised 27/03/2007) (Date of access 15/03/2008).

[2] Batini C., Scannapieca M., Data Quality – Concepts, Methodologies and Techniques, Springer, 2006

[3] Bellatrèche L., La conception physique des data warehouses, LISI/ENSMA, Paris, mai 2006.

[4] Boydens I., Les dictionnaires de données, SMALS, juin 2000.

[5] Boydens I., Data Quality : Best practices, SMALS, 2006.

[6] Brasseur C., La gestion de la qualité des données, IT-expert n°73 - juillet/août 2008.

[7] DataFlux, Methodology - Data Profiling, http://www.dataflux.com/Technology/Methodology/Data-Profiling/, juin 2008

[8] Hainaut J-L., Ingénierie des bases de données, Faculté Universitaire Notre Dame de la Paix, Namur, 2003, p5.36.

[9] Hainaut J-L., Introduction pratique à la théorie relationnelle des bases de données, Faculté Universitaire Notre Dame de la Paix, Namur, Décembre 2007.

[10] DWH-Team, Information Factory: un ensemble d'indicateurs de performance pour l'ONSS, SMALS, 2005.

[11] Inmon W., Building the Data Warehouse, Wiley, 1996.

[12] Kimball R., A dimensional modeling manifesto, DBMS Magazine, août 1997.

[13] Kimball R., The Data Warehouse Toolkit 2nd Edition, p. 8, Wiley, 2002.

[14] Menet L., Consolidation d'un modèle conceptuel de données de Master Data Management, Université de Marne-La-Vallée, 2006.

[15] Missier P., Lalk G., Verykios V., Grillo F., Lorusso T., Angeletti P., "Improving Data Quality in practice : a case study in the Italian Public Administration", Kluwer Academic Publishers, Mars 2003.

[16] Newcombe H. B., Kennedy J. M., Axford S. J., James A. P.,"Automatic linkage of vital records", Science no 130, Octobre 1959.

[17] ONSS, Notre declaration d'identité, Intranet ONSS, 14 février 2007.

[18] Olson J., Data Quality : The Accuracy Dimension, p. XV, Morgan Kaufmann, 2002.

[19] Redman T., Data Quality for the Information Age, Boston, Artech House, 1996.

[20] Savla S., Ninan M., Managing Data Quality, SetLabs Briefings Vol.3 No.4, Décembre 2005.

[21] Scannapieco M., DL3: Comparative Analysis of the Proposed Methodologies for Measuring and Improving Data Quality, Universita di Roma « La Sapienza », 2001, http://www.dis.uniroma1.it/~dq/docs.html.

[22] Shuang-lin Lee, Data Quality in organizational context : a case study, Chapel Hill, Mai 2003.

[23] Toulemonde C., Des données de Qualité, JEMM Research, 2008.

[24] Van Dromme D., Boydens I.et Bontemps Y., Data Quality : Tools, SMALS, septembre 2007.

Chapitre 8 : GLOSSAIRE

Clé technique : Egalement appelée clé de substitution ou Surrogate key, la clé technique est une clé non intelligente qui se substitue à la clé naturelle (Business Key) provenant du système opérationnel. Ajoutons que la clé de substitution est numérique. Dès lors, les performances dans les jointures sont meilleures qu'avec une clé alphanumérique.

Data Matching : Le « *Data Matching* », appelé aussi « *Data Linkage* » ou « Appariement de données », est une des dernières étapes du cycle d'amélioration de la qualité des données. Son objectif est de détecter les incohérences ou les doublons parmi les enregistrements, et d'en améliorer la qualité. Le processus de « *Data Matching* » se décompose en deux étapes : l'étape de recherche et celle de matching. La première consiste à balayer les données en identifiant les enregistrements candidats pour un matching. Sur base du résultat, le processus de matching va vérifier les candidats plus en profondeur.

Data Monitoring : Le « *Monitoring* », dernière fonctionnalité du processus d'amélioration de la qualité, consiste à mesurer et à contrôler l'évolution du niveau de qualité des données. Il permet de vérifier si les mesures correctives ont bien été appliquées à l'aide de procédure automatisée et planifiée pour fournir entre autre des rapports sur l'état de la base de données, le nombre d'erreurs détectées et corrigées, …

Data Profiling : Le « *Data Profiling* » est une des premières étapes du cycle d'amélioration de la qualité et consiste en une phase d'exploration du modèle de données à l'aide de techniques analytiques afin d'y découvrir la structure, le contenu et la qualité réelle des données, et d'y tester leur adéquation avec les métadonnées correspondantes.

Standardisation : Le processus de « Standardisation », également appelé « *Data Stadardization* », est une étape du cycle d'amélioration de la qualité. Il permet de s'assurer que toutes les données sont encodées suivant les mêmes conventions aussi bien pour les types simples que pour les types complexes, tels que les adresses ou les noms, dans un contexte multilingue. Le processus consiste à modifier les données afin qu'elles utilisent les règles métier et les standards de l'entreprise. Ces standards seront des abréviations uniformes, une orthographe correcte, des modèles de formatage,…

Doublon : Dans une base de données ou entre plusieurs bases de données, les enregistrements qui définissent une même information pour des entités ou objets du réel identique sont appelés « doublons » ou « redondance d'information ». La détection des

doublons nécessitent des procédures assez complexes capables d'associer deux enregistrements sur base d'une comparaison de caractéristiques communes, comme par exemple, les parties d'une adresse. La facilité de trouver des caractéristiques communes dépendra des similarités dans la représentation des données à l'intérieur des enregistrements à analyser.

Entrepôt de données : Le concept d' « entrepôt de données » ou « Data warehouse » a été formalisé pour la première fois à la fin des années 1980, par Bill Inmon, comme étant une structure de données orientée sujet, intégrée, historisée et non volatile. Comme les données issues du système opérationnel sont volumineuses et non organisées pour un support décisionnel, l'idée fut alors de construire une architecture exclusivement destinée à supporter des processus d'aide à la décision et en particulier les décisions stratégiques de l'entreprise. L'entrepôt de données est orienté sujet, cela signifie que les données sont rassemblées par sujet métier, par thème, dans une structure unique. L'avantage est d'avoir accès rapidement à l'ensemble des informations sur un sujet. Les données y sont non volatiles ce qui garantit qu'une même requête fournira un résultat identique dans le temps.

Gestion des métadonnées : Appelé également « *Metadata Management* », la gestion des « métadonnées » ou « méta-informations » a pour objectif de répertorier et de diffuser la documentation des applications informatiques et des modèles de données afin d'en permettre la maintenance et la réutilisation quels que soient les utilisateurs et les gestionnaires. Les « métadonnées » contiennent des informations complètes sur une table telles que la signification business, le type, la précision, le domaine de validité, les contraintes, le formatage,...